Design of an Autonomous Underwater Glider focusing on External Wing Control Surfaces and Sensor Integration

by

Cheryl Elizabeth Skibski

Bachelor of Science
Ocean Engineering
Florida Institute of Technology
2009

A thesis proposal
submitted to Florida Institute of Technology
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Ocean Engineering

Melbourne, Florida
December, 2011
We the undersigned committee hereby approve the attached thesis

Design of a Autonomous Underwater Glider focusing on External Wing Control Surfaces and Sensor Integration

by

Cheryl Elizabeth Skibski

Stephen Wood, Ph.D., P.E.
Associate Professor
Ocean Engineering Program Chair
Marine and Environmental Systems Committee Chair

Hector Gutierrez, Ph.D., P.E.
Associate Professor
Mechanical and Aerospace Engineering

Ronnal Reichard, Ph.D., P.E.
Adjunct Professor
Ocean Engineering
Marine and Environmental Systems

George A. Maul, Ph.D.
Professor and Department Chair
Marine and Environmental Systems
Abstract

Design of a Autonomous Underwater Glider focusing on External Wing Control Surfaces and Sensor Integration

by

Cheryl Elizabeth Skibski

Major Advisor: Stephen Wood, Ph.D., P.E.

This thesis incorporates the development of an inexpensive mechanically driven autonomous underwater glider. Underwater gliders travel at slow speeds acquiring high spatial and temporal resolution oceanographic data. These vehicles rely minimally on large ocean vessels and can lower costs by completing missions that last up to months at a time.

Current gliders control vertical movement by a buoyancy system and control horizontal movement by wings and internal mass movement. The glider in this thesis introduces external wing control surface flaps that deflect to provide vehicle lift. This glider is capable of operating at ocean depth up at 1000 meters, continuously acquire scientific data, and become a research platform for future glider research. Other systems on the glider include a self-powered emergency system and a Gumstix Overo Fire computer acquiring and logging sensor data.
Table of Contents

Abstract iii

List of Keywords ix

List of Figures x

List of Tables xiii

List of Acronyms xv

List of Symbols xvii

List of Equations xx

Acknowledgment xxii

Dedication xxiii

1 Introduction 1

1.1 Autonomous Underwater Gliders .......................... 1

1.2 Application ................................................. 2

1.3 Motivation ................................................. 3

2 History and Development of Underwater Gliders 6

2.1 Buoyancy-driven Floats ................................. 6
2.2 Current Underwater Glider Designs ........................................... 8
  2.2.1 Features of Current Designs ........................................... 10
  2.2.2 Seaglider ................................................................. 11
  2.2.3 Spray Glider .............................................................. 14
  2.2.4 Slocum Electric Glider ................................................. 17
  2.2.5 Slocum Thermal Glider .................................................. 18

3 Vehicle Performance .................................................................. 20
  3.1 Environmental Parameters .................................................. 20
  3.2 Performance ....................................................................... 23
  3.3 Buoyancy and Stability ...................................................... 28

4 Wings ....................................................................................... 32
  4.1 Airfoil Nomenclature ......................................................... 32
  4.2 Lift and Drag ....................................................................... 34
  4.3 Flaps .................................................................................. 37
  4.4 Wing Design ........................................................................ 39
    4.4.1 Coefficient of Lift for Finite Wings ............................... 42
    4.4.2 Coefficient of Drag for Finite Wings and Tail ............. 43
    4.4.3 Lift Force and Drag Force .......................................... 44
  4.5 Glider Wings ...................................................................... 45
    4.5.1 Construction ............................................................... 46
    4.5.2 Flap Control ............................................................... 48
    4.5.3 Wing Mounting .......................................................... 49
    4.5.4 Tail Mounting ............................................................. 50

5 Fairing ...................................................................................... 52
6 Pressure Housing

6.1Yielding ................................................. 55
6.2Buckling .................................................. 56
6.3General Instability ................................. 57
6.4Compressibility ........................................ 57
6.5End Plate Stress ................................. 58
6.6Design .................................................... 60

7 Gumstix Overo Fire ................................. 62

7.1Python Programming Language ....................... 63
7.2Gumstix Overo Fire Computer and Tobi Expansion Board ...... 64
7.3Global Positioning System (GPS) ...................... 67
7.4Inertial Measurement Unit (IMU) ..................... 68
7.5Compass .................................................. 70
7.6Abort Signal to Emergency System ................. 71
7.7Flap Deflection Control ............................. 72
7.7.1PWM Generation ........................................ 74
7.7.2Flap Deflection Test Results ..................... 77

8 Vehicle Control ........................................ 78

9 Emergency System .................................... 81

9.1Microcontroller PIC18LF45k80 .......................... 81
9.2Sensors ................................................. 84
9.2.1Temperature and Humidity Sensor ............... 84
9.2.2Pressure Sensor ........................................ 85
9.2.3Runtime Timer .......................................... 86
G  PWM Register Calculations  163

H  Cost List  165

I  Datasheets  168
List of Keywords

 Autonomous Underwater Vehicle
 buoyancy-driver underwater floats
 Spray Glider
 Seaglider
 Slocum Glider
 Glider
 wing
 tail
 control surfaces
 wing control
 live hinge
 flaps
 fairing
 buoyancy system
 pressure housing
 sensor integration
 Gumstix Overo Fire
 Tobi Expansion Board
 pulse-width modulation
 Atomic 6 DOF Inertial Measurement Unit
 Tilt-compensated compass
 servo motor
 Global Positioning System
 Python
 emergency drop weight
List of Figures

1.1 Autonomous Underwater Vehicle and Glider .......................... 3
2.1 Diagram of Ocean Floats .............................................. 7
2.2 Path of Ocean Floats .................................................. 7
2.3 Seaglider ............................................................... 11
2.4 Seaglider Path ......................................................... 13
2.5 Spray Glider ............................................................ 15
2.6 Spray Glider Path ....................................................... 16
2.7 Spray Glider Schematic ............................................... 17
2.8 Slocum Glider .......................................................... 17
2.9 Slocum Thermal Glider Diagram ...................................... 19
3.1 Density ................................................................. 21
3.2 Temperature ............................................................ 22
3.3 Salinity ................................................................. 22
3.4 Dive Profile ............................................................. 24
3.5 Forces on the Glider .................................................... 24
3.6 Flap Deflection vs. Glide Angle .................................... 27
3.7 Glider Path ............................................................. 28
3.8 Simple Buoyancy Engine ............................................. 30
4.1 Airfoil Nomenclature ................................................... 33
7.9  Servo Motor ................................................. 73
7.10 Servo Motor Board ....................................... 73
7.11 Pulse Width Modulation (PWM) ......................... 74

8.1  Glider Electronics ........................................ 78

9.1  Flow Diagram for the Emergency System ............... 82
9.2  Emergency System Board ................................ 83
9.3  Microcontroller and Programmer ....................... 83
9.4  Temperature and Humidity Sensor ...................... 85
9.5  Barometric Pressure Sensor ............................. 86
9.6  Drop Weight Mechanism ................................ 88
9.7  RF Modem ................................................ 89
9.8  Datalogger ................................................ 90

10.1  Star Oddi Conductivity Temperature Depth (CTD) Sensor . . 92

11.1  Hyperbaric Chamber .................................... 94
11.2  Wet Connectors .......................................... 95
11.3  Sealed Pressure Housing ................................ 95
11.4  Emergency System Test ................................. 96
11.5  Radio Frequency (RF) Receiver Board ................ 97
11.6  Wing Flap Deflection Results ......................... 99
11.7  Servo Angle and Flap Deflection of Right Wing ........ 100
11.8  Servo Angle and Flap Deflection of Left Wing .......... 100
List of Tables

1 List of Acronyms ................................................. xv
2 List of Symbols ................................................ xvii

2.1 Seaglider Specifications ................................. 12
2.2 Spray Glider Specifications ............................... 15
2.3 Slocum Glider Specifications ............................. 18

3.1 Glider Performance ......................................... 26

4.1 Wing Results ................................................ 45
4.2 Tail Results .................................................. 46

5.1 Fairing Results .............................................. 53

6.1 Pressure Hull Design Data ............................... 61
6.2 Dimensions for Park O-Ring 2-261 .................... 61

7.1 Overview of Sensors ...................................... 63
7.2 Overo Fire 40-Pin Header on Tobi Expansion Board .. 66
7.3 RMC-Recommended Minimum Data ..................... 67
7.4 IMU Output .................................................. 69
7.5 Timer Address Summary ................................. 75
7.6 PWM Output Definitions ................................. 75
7.7 Angle and TMAR value .................................... 77
## List of Acronyms

Table 1: List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC or A/D</td>
<td>Analog-to-digital Converter</td>
</tr>
<tr>
<td>ALACE</td>
<td>Autonomous Lagrangian Circulation Explorer</td>
</tr>
<tr>
<td>ASCII</td>
<td>American Standard Code for Information Interchange</td>
</tr>
<tr>
<td>AUV</td>
<td>Autonomous Underwater Vehicle</td>
</tr>
<tr>
<td>CAD</td>
<td>Computer-aided Design</td>
</tr>
<tr>
<td>CTD</td>
<td>Conductivity, Temperature, Depth</td>
</tr>
<tr>
<td>FTDI</td>
<td>Future Technology Devices International</td>
</tr>
<tr>
<td>GMT</td>
<td>Greenwich Mean Time</td>
</tr>
<tr>
<td>GPIO</td>
<td>General Purpose Input/Output</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>HDPE</td>
<td>High-density polyethylene</td>
</tr>
<tr>
<td>I²C</td>
<td>Inter-Integrated Circuit</td>
</tr>
<tr>
<td>IMU</td>
<td>Inertial Measurement Unit</td>
</tr>
<tr>
<td>MSSP</td>
<td>Master Synchronous Serial Port</td>
</tr>
<tr>
<td>NACA</td>
<td>National Advisory Committee for Aeronautics</td>
</tr>
<tr>
<td>NASA</td>
<td>National Aeronautics and Space Administration</td>
</tr>
<tr>
<td>NMEA</td>
<td>National Marine Electronics Association</td>
</tr>
<tr>
<td>OMAP</td>
<td>Open Multimedia Application Platform</td>
</tr>
<tr>
<td>ONR</td>
<td>Office of Naval Research</td>
</tr>
<tr>
<td>ORBCOMM</td>
<td>Orbital Communications</td>
</tr>
<tr>
<td>PIC</td>
<td>Peripheral Interface Controller</td>
</tr>
</tbody>
</table>

*continued on next page*
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PWM</td>
<td>Pulse-width Modulation</td>
</tr>
<tr>
<td>RAM</td>
<td>Random-access Memory</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>RX</td>
<td>Receive Line</td>
</tr>
<tr>
<td>RXO</td>
<td>Receive Line Output</td>
</tr>
<tr>
<td>SD</td>
<td>Secure Digital</td>
</tr>
<tr>
<td>SPI</td>
<td>Serial Peripheral Interface</td>
</tr>
<tr>
<td>TX</td>
<td>Transmit Line</td>
</tr>
<tr>
<td>TXI</td>
<td>Transmit Line Input</td>
</tr>
<tr>
<td>UART</td>
<td>Universal Asynchronous Receiver/Transmitter</td>
</tr>
<tr>
<td>USART</td>
<td>Universal Synchronous Asynchronous Receiver/Transmitter</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Serial Bus</td>
</tr>
<tr>
<td>UTC</td>
<td>Universal Time Clock</td>
</tr>
</tbody>
</table>
## List of Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Cross Sectional Area of Ring Reinforcement</td>
</tr>
<tr>
<td>a</td>
<td>Lift slope coefficient for finite wing</td>
</tr>
<tr>
<td>a₀</td>
<td>Lift slope coefficient for infinite wing</td>
</tr>
<tr>
<td>aᵣ</td>
<td>Lift slope coefficient for finite wing (Raymer Method)</td>
</tr>
<tr>
<td>Aᵦ</td>
<td>Fairing cross-sectional Area</td>
</tr>
<tr>
<td>A_C</td>
<td>Characteristic Area</td>
</tr>
<tr>
<td>A_f</td>
<td>Area of frame</td>
</tr>
<tr>
<td>A_T</td>
<td>Tail Planform Area</td>
</tr>
<tr>
<td>A_W</td>
<td>Wing Planform Area</td>
</tr>
<tr>
<td>Accel Sensitivity</td>
<td>Sensitivity of the Accelerometer</td>
</tr>
<tr>
<td>ADCVal</td>
<td>Accelerometer Value Read from the IMU</td>
</tr>
<tr>
<td>AR</td>
<td>Aspect Ratio</td>
</tr>
<tr>
<td>B</td>
<td>Buoyant Force</td>
</tr>
<tr>
<td>b_f</td>
<td>Width of frame web</td>
</tr>
<tr>
<td>b</td>
<td>Wing span</td>
</tr>
<tr>
<td>c_l</td>
<td>Coefficient of Lift of Infinite Wing</td>
</tr>
<tr>
<td>c_d</td>
<td>Coefficient of Drag of Infinite Wing</td>
</tr>
<tr>
<td>c_df</td>
<td>Skin friction drag coefficient</td>
</tr>
<tr>
<td>c_dp</td>
<td>Pressure drag coefficient</td>
</tr>
<tr>
<td>c_dw</td>
<td>Wave drag coefficient</td>
</tr>
<tr>
<td>C_DB</td>
<td>Coefficient of Drag of the Fairing</td>
</tr>
</tbody>
</table>

*continued on next page*
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(C_{DT})</td>
<td>Coefficient of Drag of the Tail</td>
</tr>
<tr>
<td>(C_{DW})</td>
<td>Coefficient of Drag on the Wings</td>
</tr>
<tr>
<td>(C_{FB})</td>
<td>Coefficient of Frictional Drag of the Fairing</td>
</tr>
<tr>
<td>(C_{LW})</td>
<td>Coefficient of Lift on the Wings</td>
</tr>
<tr>
<td>(d)</td>
<td>Fairing diameter (Raymer Method)</td>
</tr>
<tr>
<td>(D)</td>
<td>Diameter</td>
</tr>
<tr>
<td>(DC)</td>
<td>Duty Cycle</td>
</tr>
<tr>
<td>(E)</td>
<td>Modulus of Elasticity</td>
</tr>
<tr>
<td>(e)</td>
<td>Oswald Efficiency Factor</td>
</tr>
<tr>
<td>(f)</td>
<td>frequency</td>
</tr>
<tr>
<td>(F_L)</td>
<td>Force of Lift</td>
</tr>
<tr>
<td>(F_D)</td>
<td>Force of Drag</td>
</tr>
<tr>
<td>(FCLK)</td>
<td>Clock Frequency</td>
</tr>
<tr>
<td>(\text{GyroRate})</td>
<td>Gyro Rate</td>
</tr>
<tr>
<td>(\text{GyroADCValue})</td>
<td>Gyro Reading from the IMU sensor</td>
</tr>
<tr>
<td>(\text{Gyro Sensitivity})</td>
<td>Sensitivity of the Gyro Sensor on the IMU</td>
</tr>
<tr>
<td>(I)</td>
<td>Moment of Inertia of Ring Reinforcement</td>
</tr>
<tr>
<td>(I_l)</td>
<td>Moment of Inertia for Instability Calculations</td>
</tr>
<tr>
<td>(L_f)</td>
<td>Length between Structural stiffeners</td>
</tr>
<tr>
<td>(L)</td>
<td>Length</td>
</tr>
<tr>
<td>(m)</td>
<td>mass</td>
</tr>
<tr>
<td>(M)</td>
<td>Bending Moment</td>
</tr>
<tr>
<td>(M_c)</td>
<td>Bending Moment at loading center</td>
</tr>
<tr>
<td>(M_N)</td>
<td>Mach Number</td>
</tr>
<tr>
<td>(M_{ra})</td>
<td>Bending Moment at reaction</td>
</tr>
<tr>
<td>(n)</td>
<td>The integer number resulting in the lowest value of (P_{cr}), (n=2)</td>
</tr>
<tr>
<td>(P_b)</td>
<td>Buckling Pressure</td>
</tr>
<tr>
<td>(P_{cr})</td>
<td>Critical Pressure</td>
</tr>
<tr>
<td>(P_y)</td>
<td>Yield Pressure</td>
</tr>
<tr>
<td>(q)</td>
<td>Maximum pressure at designed depth</td>
</tr>
<tr>
<td>(R)</td>
<td>Radius of average diameter of the shell</td>
</tr>
<tr>
<td>(R_x)</td>
<td>Force in the x-direction</td>
</tr>
</tbody>
</table>

continued on next page
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>Radius of the diameter of the end plate not constrained</td>
</tr>
<tr>
<td>Re</td>
<td>Reynolds Number</td>
</tr>
<tr>
<td>$S_{Exposed}$</td>
<td>Wing Planform area exposed (Raymer Method)</td>
</tr>
<tr>
<td>$S_{Ref}$</td>
<td>Wing Planform area reference (Raymer Method)</td>
</tr>
<tr>
<td>t</td>
<td>Thickness</td>
</tr>
<tr>
<td>T</td>
<td>Signal Period</td>
</tr>
<tr>
<td>$T_{Surface}$</td>
<td>Time to surface after a complete dive cycle</td>
</tr>
<tr>
<td>V</td>
<td>Velocity of vehicle</td>
</tr>
<tr>
<td>$V_{H}$</td>
<td>Horizontal Component of Velocity</td>
</tr>
<tr>
<td>$V_{V}$</td>
<td>Vertical Component of Velocity</td>
</tr>
<tr>
<td>$V_{zeroG}$</td>
<td>Voltage at Zero g</td>
</tr>
<tr>
<td>Vol</td>
<td>Volume</td>
</tr>
<tr>
<td>W</td>
<td>Weight</td>
</tr>
<tr>
<td>$y_c$</td>
<td>Flat plate deflection at the center</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Angle of Attack</td>
</tr>
<tr>
<td>$\alpha_0$</td>
<td>Angle of Attack for Zero Lift</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Kinematic Viscosity</td>
</tr>
<tr>
<td>$\delta V$</td>
<td>Change in volume</td>
</tr>
<tr>
<td>$\theta$</td>
<td>Angle of Descent</td>
</tr>
<tr>
<td>$\nu$</td>
<td>Poisson’s Ratio for hull material</td>
</tr>
<tr>
<td>$\xi$</td>
<td>Hull Surface to Neutral Axis of Ring Reinforcement Distance</td>
</tr>
<tr>
<td>$\rho$</td>
<td>Density of Fluid</td>
</tr>
<tr>
<td>$\tau$</td>
<td>Duration the pulse-width modulated signal is high</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Stress</td>
</tr>
<tr>
<td>$\sigma_c$</td>
<td>Stress at center</td>
</tr>
<tr>
<td>$\sigma_{ra}$</td>
<td>Stress of reaction</td>
</tr>
<tr>
<td>$\sigma_y$</td>
<td>Yield Strength</td>
</tr>
<tr>
<td>$\wedge_{LE}$</td>
<td>Leading Edge Sweep Angle in degrees</td>
</tr>
</tbody>
</table>
# List of Equations

3.1 Glide Angle .................................................. 25  
3.3 Glide Range .................................................. 25  
3.4 Time to Surface .............................................. 26  
3.6 Buoyancy ..................................................... 29  
3.7 Change in Volume ............................................ 29  
3.8 Density ....................................................... 30  
3.9 Vehicle Weight ............................................... 30  
4.1 Reynolds Number for Wings ................................. 39  
4.2 Aspect Ratio .................................................. 41  
4.4 Coefficient of Lift for Finite Wing ....................... 42  
4.5 Lift Slope Coefficient of an Infinite Wing ............. 42  
4.6 Lift Slope Coefficient for a Finite Wing ................ 42  
4.9 Coefficient of Drag for Finite Wing and Tail .......... 43  
4.10 Coefficient of Drag for a Finite Wing ................. 43  
4.11 Coefficient of Drag for a Tail ............................ 44  
4.12 Lift Force ................................................... 44  
4.13 Drag Force .................................................. 44  
5.1 Reynolds Number for the Fairing ......................... 52  
5.2 Coefficient of Frictional Drag ............................. 52  
5.3 Coefficient of Drag .......................................... 52
Acknowledgment

I would like to acknowledge the following faculty, co-workers, and family who has made this research and thesis possible.

- Dr. Stephen Wood
- The Underwater Technology Lab
- Dr. Ronnal Reichard and Structural Composites Inc.
- Dr. Hector Gutierrez
- The Department of Aerospace Engineering
- Center for Corrosion and Biofouling Control (CCBC)
- Tonya Mitchell and Linda Lundstedt in the Department of Marine and Environmental Systems office
- Family and Friends

Without them, I could never complete this.
Dedication

I would like to dedicate this thesis to my mom, my dad, and my husband to be.
Chapter 1

Introduction

1.1 Autonomous Underwater Gliders

Autonomous underwater gliders make up a new and emerging field of underwater vehicles. These vehicles glide through the ocean at low speeds using changes in buoyancy to ascend and descend and wings to generate lift to propel the vehicle forward. Gliders conduct long duration missions to acquire and communicate scientific data to researchers throughout the world without direct involvement of ships or personnel. Due to the ability to glide at slow speeds, gliders have longer deployment capabilities than standard powered autonomous underwater vehicles, making gliders extremely valuable for long term oceanographic sensing missions covering a large area of the ocean. Understanding the components of a glider is essential for future improvements of underwater gliders.

Overall, this thesis introduces gliders and details the design of a mechanically-driven autonomous underwater glider. It covers the historical development of underwater gliders as well as detailing the important characteristics of current gliders that are key to glider performance. This thesis covers vehicle control,
sensor integration, and a self-powered vehicle emergency system.

1.2 Application

Protecting the ocean is essential to future generations. Acquiring data directly at depth with high temporal and spatial resolution is necessary in order to continuously monitor and analyze ocean trends, learn about ocean processes, predict climate change, and acquire physical, biological, and oceanographic data [1]. Currently, gathering oceanographic data is performed by remote sensing, deploying oceanographic instruments aboard ships, and the use of autonomous underwater vehicles. Since remote sensing is limited to the ocean surface and deploying instruments aboard ships is very costly and time consuming, autonomous underwater vehicles are used for data collection. The autonomous underwater vehicle (AUV) shown in Figure 1.1(a) is a propulsion driven underwater vehicle that uses a propeller to move. Similarly, the autonomous underwater glider in Figure 1.1(b) is a low power AUV that uses buoyancy changes for vertical movement and wings for horizontal movement to propel the glider forward. A glider is an easy and affordable way to acquire continuous data such as conductivity, temperature, depth, dissolved oxygen, and turbidity throughout the water column. Because gliders are small and low power vehicles, they are able to log data up to months at a time and depend minimally on large ocean vessels.

During a mission, gliders can operate in harsh deep ocean and coastal aquatic environments. They are able to circle an area of interest, wait at the bottom of a glide path for long periods of time, and quietly move undetected. Gliders have little impact to the environment and provide minimum threat to sea creatures
due to their slender body and slow movement through the water. The characteristics and feasibility of gliders has led to a high interest in glider research at Florida Institute of Technology for oceanographic data collection, experience, and testing. This thesis researches and tests a new component for underwater gliders.

1.3 Motivation

The new and unique feature of underwater gliders discussed in this thesis is a set of external control surfaces on the wings, also known as flaps. These external
surfaces move up and down to aid in pitch, roll, and horizontal movement control of the glider. The following points emphasize this new feature and additional motivation for this thesis.

1. **Live Hinge Flap Design**

   Most aircraft use flaps with a hinge point and valuable components of the hinge exposed. Seawater contains microorganisms and algae that can get caught in the hinge areas around flaps. External moving parts and the use of materials susceptible to corrosion increases biofouling and decreases movement. The bendable live hinge on this glider is made out of a thin, flexible, low-modulus fiberglass.

2. **Create a underwater glider research and vehicle foundation at Florida Institute of Technology**

   Undergraduate and graduate students ranging from all engineering fields will learn about gliders, be able to integrate instruments and sensors, and learn how to deploy and recover the vehicle.

3. **Create a low cost small size glider to deploy from small boats and acquire necessary scientific data**

   This glider is developed at a low cost by using resources on campus, scrap material, and low cost electronic sensors. The current gliders discussed in Chapter 2 have an estimated construction cost ranging from $25,000 to $60,000 and a refueling cost ranging from $800 to $3,850 [1]. This glider is estimated to be completed in a cost range equal to or less than the refueling cost of the current gliders.

   This project is supported by faculty who have experience in this subject of engineering either in research, education, or in the workplace. The project is
supported and funded by the Florida Institute of Technology Department of Marine and Environmental Systems.
Chapter 2

History and Development of Underwater Gliders

The concept of underwater gliders was pictured first by Henry Stommel in 1989. He envisioned a group of underwater floats called “Slocums” capable of moving vertically and horizontally in the water returning observations of the changing state of the ocean [3]. His vision led to the design of buoyancy-driven floats which turned into the development of underwater gliders.

2.1 Buoyancy-driven Floats

The Autonomous Lagrangian Circulation Explorer (ALACE) buoyancy-driven float shown in Figure 2.1 resulted from Stommel’s idea. According to Figure 2.2, this float alternates between drifting and profile modes to acquire data directly at depth to gain a understanding of ocean dynamics.
Figure 2.1: Diagram of Ocean Floats [4]

Figure 2.2: Path of Ocean Floats [4]
The ALACE float, along with the Profiling Autonomous Lagrangian Circulation Explorer (PALACE) and Salinity Profiling Autonomous Lagrangian Circulation Explorer (S-PALACE) are part of the Argo network, which is a global array of profiling floats that began deployments in 2000, totaling 3308 floats as of July 24, 2011. These floats stay out for months capturing data with temperature, salinity, and current sensors and sending data back to shore through a satellite link to the Argo network [4].

Current glider buoyancy systems originated from a method similar to the buoyancy system on the ALACE float, known as the Leduc ballast system. This system contains an extruding piston that moves in and out of the bottom of the float by a motor driven lead screw system. With a known volume change, the float can change from negatively to positively buoyant and move throughout the ocean. The float uses an ORBCOMM satellite uplink to transmit data, receive instructions, and allow scientists to obtain semi real-time observations to estimate the climate system and observe the physical state of the ocean [5].

Buoyancy-driven floats have the advantage of mobility and low manufacturing costs. The disadvantage for floats is the lack of position control. Currents have a major influence on float location and the float is unable correct its position, speed, and direction. Due to the need of control, underwater gliders were designed to provide a control system, navigation algorithm, and a low drag vehicle with appendages to observe all areas of the ocean.

2.2 Current Underwater Glider Designs

Current gliders include the University of Washington’s Applied Research Laboratory and iRobot’s Seaglider, Teledyne Webb Research’s Slocum Elec-
tric and Thermal Glider, and Scripps Oceanographic Institution and Bluefin Robotics Corporation’s Spray Glider. The following timeline states the design, development, and first sea travels of the three gliders.

• **1970**: Sound Fixing And Ranging (SOFAR) floats were developed to measure ocean currents [6].

• **1990**: Origins of the Argo network for buoyancy-drive floats began with the ALACE float [4].

• **1991**: Spray glider performed preliminary lake tests [3]. A prototype was designed and tested for the Slocum battery powered glider in Wakulla Springs, FL using an autopilot and flight recorder [7].

• **1995**: Webb Research Corporation tested the hull of the Spray glider [3]. A vertical profiling vehicle powered by thermal propulsion was deployed in the Sargasso Sea [7].

• **1997**: Design of the Spray glider began [3].

• **1998**: A glider and thermal engine were deployed at Seneca Lake, NY with a thermocline [7].

• **1999**: First Spray sea trail with the Autonomous Ocean Sensing Network (AOSN) experiment in the Monterey Underwater Canyon. Spray was recovered 11 days later with 182 logged temperature and conductivity profiles [3].

  Seaglider Port Susan Deployment to the Possession Sound with logged GPS fixes and dead reckoning displacements [8]
• 2004: Seaglider’s launch in the North Pacific Ocean and recovery at the shore of Kauai, Hawaii [2].

2.2.1 Features of Current Designs

Sections 2.2.2-2.2.5 describe in brief the current underwater gliders. The following list are features similar in all three gliders.

• Design for long duration ocean sensing missions

• Travel at a slow speed which is crucial for oceanographic observations

• Fairing shapes that decrease vehicle drag, fixed wings, and a fixed tail

• Relatively the same size, shape, and weight measuring about 2 meters in length and 50 kg in weight

• Antenna protruding from areas that do not increase vehicle drag

• Internal mass movement to control pitch and roll

• Ballast system that uses a hydraulic pump to move oil between the external bladder and internal reservoir similar to buoyancy driven floats

• Drop weight system in case of emergency
2.2.2 Seaglider

The Seaglider displayed in Figure 2.3 was built, tested, and deployed by the University of Washington’s Applied Physics Laboratory and currently sold by iRobot. The free-flooded hydrodynamic low drag fairing is made out of fiberglass to provide support for the wings, rudder, and the aluminum pressure housing [2]. The airfoil section of the body and passive wings provide the highest overall lift to drag ratio for the low Reynolds number experienced with underwater gliders. The NACA 16-006 airfoil section of the wings provide laminar flow for up to 80% of the chord [8].
As seen in Table 2.1, the speed of the vehicle is greater than or equal to 0.25 m/s in order to overcome ocean currents and cover a large ocean range. The Seaglider can travel at glide slopes from 16 to 45°. Greater glide slopes result in a shorter range and lower glide slopes result in the longer range ideal for oceanographic surveying [2].

Table 2.1: Seaglider Specifications [2]

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>52 kg</td>
</tr>
<tr>
<td>Hull Diameter</td>
<td>30 cm</td>
</tr>
<tr>
<td>Vehicle Length</td>
<td>1.8 m</td>
</tr>
<tr>
<td>Wing Span</td>
<td>1 m</td>
</tr>
<tr>
<td>Maximum Depth</td>
<td>1000 m</td>
</tr>
<tr>
<td>Speed</td>
<td>0.25 m/s</td>
</tr>
<tr>
<td>Range</td>
<td>4600 km</td>
</tr>
<tr>
<td>Glide Angle</td>
<td>16-45°</td>
</tr>
</tbody>
</table>

The Seaglider completes the sawtooth shown in Figure 2.4 using a computer that decides on the bearing and glide slope to the target position based on the vehicle's current position. The glide terminates with a vehicle pitch of 45° at the surface to expose the antenna located at the aft of the vehicle to the sky. The waterproof antenna contains a Global Positioning System (GPS) and a wireless modem that transmits information to shore and downloads a new file of instructions [8].

The Seaglider has the ability to glide up to months at a time by adjusting the volume in the buoyancy control system to be smaller or larger than the mass of seawater. For flight control, the vehicle controls the center of gravity relative to the center of buoyancy. The center of gravity and the center of buoyancy both change throughout a dive due to vehicle movement. The Seaglider moves a high voltage battery pack fore and aft to provide pitch control, and rolls the battery
Figure 2.4: Seaglider Path [2]

pack left and right for roll control instead of using external control surfaces [8].

The buoyancy system is similar to the design seen on the buoyancy-driven floats in Section 2.1. This system includes a boost pump with a pump and motor manufactured by HydroLeduc [8]. The internal reservoir made of a Bellowfram Diaphragm [8] contains a vacuum for oil to bleed from the external to internal reservoir. Moving the oil from the internal to external reservoir is done through an axial pump mechanism [8].

Sensors on the Seaglider include a conductivity cell, thermistor, dissolved oxygen sensor, pressure sensor, and acoustic transducer. Two battery packs
of lithium thionyl chloride D-Cell batteries provide high and low voltage to the vehicle. The navigation software provides a dead reckoning position during the dive cycles and corrected with actual surface positions by the use of the Kalman filter. An acoustic transducer located on the front of the vehicle is used for locating and tracking the vehicle [8].

Currently, the Seaglider is sold by iRobot and used for multiple missions of profiling the ocean and has been monitoring the after effects of the Deepwater Horizon oil spill in May 2010. Seaglider milestones include the first glider to complete a mission greater than 3800 km and the first to complete a multi-glider mission. This glider also completed a mission greater than 6 months in March 2005, which consisted of up to 191 days of runtime and up to 600 dives of 1000 meter depth [2]. The limiting factor on duration includes battery life and the longevity of the on board components to the exposure of seawater for a large amount of time.

2.2.3 Spray Glider

The Spray Glider in Figure 2.5 was developed under the Office of Naval Research (ONR) with support from Scripps Institution of Oceanography and currently sold by Bluefin Robotics Corporation. This autonomous underwater vehicle uses battery power to pump hydraulics through the buoyancy control system to change volume and power while gliding.
According to the vehicle specifications in Table 2.2, the Spray Glider is designed to dive up to 1500 meters at speeds up to 25 cm/s. It uses a see-saw path shown in Figure 2.6, controlling the dive angle during ascent and descent by moving battery packs of Lithium DD cell batteries forward and backward to change pitch and rotating battery packs to change roll. At the surface, the glider rotates 90° to utilize one of the GPS and Iridium antennas located on the wing to transmit data to shore and allow for operators to change the course of a mission. The wings have a thicker airfoil shape to allow for better stiffness and protection for the enclosed antenna [3].

Table 2.2: Spray Glider Specifications [9]

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>51.8 kg</td>
</tr>
<tr>
<td>Hull Diameter</td>
<td>0.2 m</td>
</tr>
<tr>
<td>Vehicle Length</td>
<td>2.0 m</td>
</tr>
<tr>
<td>Wing Span</td>
<td>1.1 m</td>
</tr>
<tr>
<td>Maximum Depth</td>
<td>1500 m</td>
</tr>
<tr>
<td>Speed</td>
<td>0.25-0.30 m/s</td>
</tr>
<tr>
<td>Range</td>
<td>4700-3500 km</td>
</tr>
<tr>
<td>Glide Angle</td>
<td>19-25°</td>
</tr>
</tbody>
</table>
Figure 2.6: Spray Glider Path [9]

Figure 2.7 shows the instruments aboard the Spray Glider. These include conductivity and temperature sensors located on the top of the flooded tail section, fluorometer, backscatter sensor, altimeter, and an acoustic pinger to allow for underwater tracking. The emergency system consists of a drop weight that allows the vehicle to surface in case of failure [3].

Variables involved in the glide control algorithm include heading, roll, pitch, pressure, and altitude. Sensors measuring these variables include a compass, pressure gage, and acoustic altimeter. The control loop for pitch control is proportional with a low gain and the heading control loop includes a proportional and integral term to account for errors. A simple navigation algorithm computes distance and heading from the GPS fix to the desired waypoint [3].
2.2.4 Slocum Electric Glider

The Slocum Electric Glider shown in Figure 2.8 was developed by Teledyne Webb Research and named after Joshua Slocum, the first person to circumnavigate the globe [7].
The hull of the electric glider is made out of Aluminum 6061-T6 and the fixed wings are made out of composite. Pitch and roll control is achieved by translating and rotating the main battery pack. The yaw moment is achieved by using a rudder on the tail and mounting the wings aft of the center of buoyancy. The antenna is located on the tail of the vehicle for communication when surfaced [7].

The vehicle is powered by alkaline batteries, deploying from up to 30 days at a 600 to 1500 km range as shown under the electric glider portion of Table 2.3. Sensors on board include GPS, compass, altimeter, RF modem, optical sensor, conductivity temperature depth (CTD) sensor, and an oxygen sensor [10]. The Slocum electric glider has a buoyancy engine consisting of a single stroke pump that pushes water in and out of the nose of the vehicle. An air pump is used to inflate the air bladder using air that is in the interior of the hull. This provides additional needed buoyancy when the vehicle is on the surface of the water [10].

Table 2.3: Slocum Glider Specifications [10]

<table>
<thead>
<tr>
<th></th>
<th>Electric</th>
<th>Thermal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>52 kg</td>
<td>60 kg</td>
</tr>
<tr>
<td>Hull Diameter</td>
<td>21.3 cm</td>
<td>21.3 cm</td>
</tr>
<tr>
<td>Length</td>
<td>1.5 m</td>
<td>1.5 m</td>
</tr>
<tr>
<td>Maximum Depth</td>
<td>4-200 m (coastal) 1000 m (deep)</td>
<td>1200 m</td>
</tr>
<tr>
<td>Speed</td>
<td>0.40 m/s</td>
<td>0.40 m/s</td>
</tr>
<tr>
<td>Range</td>
<td>1500 km (one battery charge)</td>
<td>40,000 km (5 year period)</td>
</tr>
</tbody>
</table>

2.2.5 Slocum Thermal Glider

The Slocum thermal glider, similar to the electric glider according to Table 2.3 uses the environmental thermal energy to propel the vehicle vertically in the
water. Also developed by Teledyne Webb Research, this glider’s engine uses the melting and freezing of wax to harvest energy from the ocean and drive the buoyancy system. The thermal glider also moves oil into a flexible bladder in the nose cone for pitch control. Figure 2.9 displays an image of the Slocum Thermal glider and labels the features apparent in both the electric and thermal glider. The main difference are the thermal engine tubes underneath the vehicle protruding past the antenna fin.

Figure 2.9: Slocum Thermal Glider Diagram [11]

The feasibility of the thermal glider includes a long projected endurance of up to 5 years, dives down to a maximum depth of 1200 m, and the ability to operate in 65% of the world’s oceans. The disadvantage of this thermal engine is that it can only be used when a thermal gradient is available [7].
Chapter 3

Vehicle Performance

The ocean environment has a large impact on the performance of gliders. Vehicle navigation depends mainly on ocean currents, wings and glider stability depends largely on characteristics of the surrounding water, and the buoyancy system depends immensely on water density. This chapter details environmental parameters and explains equations to predict glider performance.

3.1 Environmental Parameters

The Practical Salinity Scale (PSS-78) states that the equation of seawater salinity calculation is affected by temperature, density, and pressure and is still archived in national databases [12]. According to [5], the equation $\rho=\rho(S,T,P)$, displays density as a function of salinity, temperature, and pressure. This becomes clearer in Figure 3.1 to Figure 3.3, which displays data from Rutger’s University Coastal Engineering Lab acquired from the Slocum Electric Glider RU1. The New Jersey Self Observing System (NJSOS) displays this data from August 22, 2004 at 23:40:23 GMT - August 30, 2004 at 12:16:30 GMT [11].
Pressure is measured by $P = \rho gh$, where $\rho$ is the density of seawater, varying between 1020-1030 kg/m$^3$ in Figure 3.1. At sea level pressure is 101 kPa (14.7 psi), and doubles every atmosphere, or every 10.1 m (33 ft). As depth increases, temperature decreases in Figure 3.2 due to the movement closer to the center of the earth and the movement away from sunlight. Salinity changes due to changes in temperature, density, and movement toward water with higher salt content.

Figure 3.1: Density [11]
Figure 3.2: Temperature [11]

Figure 3.3: Salinity [11]
The viscosity of seawater and the Reynolds Number are critical values when determining flow around underwater gliders. Viscosity defines how a fluid flows and characterizes how the fluid behaves as it comes into contact with an object. Kinematic viscosity, the term used in the Reynolds Number calculation is the ratio of the absolute viscosity to the fluid density. The Reynolds number is the ratio between the dynamic and viscous forces of a fluid and determines if the flow is laminar or turbulent around the vehicle and where flow separation occurs [13]. These parameters all effect the performance of the glider in one way or another. The water density impacts the glider buoyancy and change in volume for a steady glide. The Reynolds number predicts the flow around the airfoil, and determines the lift and drag of the wings which can have a major impact on horizontal movement.

### 3.2 Performance

Autonomous underwater gliders ascend and descend using buoyancy changes to travel vertically and wings to travel horizontally. Figure 3.4 shows a basic dive profile for underwater gliders. Major factors such as the hydrodynamic forces and moments of the body and wings contribute to the motion of the glider.

The forward traveling velocity is affected by drag and lift and both are important factors for determining vehicle power and horizontal translation of the vehicle. Drag and Lift forces are calculated in Chapter 4. The lift force on the wings, drag force on the body, drag force on the wings, and drag force on the tail shown in Figure 3.5 all contribute to the vehicle performance. The lift forward force balances the backward force of drag and the weight of the vehicle is balanced by lift and drag. Gliders are designed to have minimum appendages
Figure 3.4: Dive Profile

Figure 3.5: Forces on the Glider
that may cause an increase drag on the vehicle. An example of a contributing factor of drag include antennas projecting out of the glider. To prevent this, antennas for global positioning and satellite communication are often implanted into the tail or wing.

Summing up the lift and drag forces at a desired velocity, the glide angle can be determined. The glide angle, $\theta$, in Equation 3.1 and also shown in Figure 3.5 is the angle between the velocity vector and the horizontal line. Each force component in Figure 3.5 is composed of an area term and a coefficient term. Variables $A_W$, $A_T$, and $A_B$ are the wing planform area, the tail planform area, and the fairing cross-sectional area, respectively. $C_{DW}$, $C_{DB}$, $C_{DT}$, and $C_{LW}$ are the coefficients of drag for the wing, body, and tail and the coefficient of lift of the wing. These variables are calculated in Chapters 4 and 5.

$$\tan \theta = \frac{A_W C_{DW} + A_B C_{DB} + A_T C_{DT}}{A_W C_{LW}}$$  \hspace{1cm} (3.1)

The velocity vector has a horizontal ($V_H$) and vertical component ($V_V$). Equation 3.2 shows equations to solve the horizontal and vertical components using the actual velocity as $V$.

$$V_H = V \cos \theta$$  \hspace{1cm} (3.2)

$$V_V = V \sin \theta$$

The horizontal component of velocity is used to determine the range of the dive in Equation 3.3, defined as the horizontal distance on the surface of the ocean between points where the glider surfaces.

$$Range = T_{Surface} V_H$$  \hspace{1cm} (3.3)
The duration of a dive is critical for any autonomous underwater vehicle operation. The time it takes for each dive cycle is calculated in Equation 3.4 using the the maximum dive depth of the glider.

\[
T_{\text{Surface}} = 2 \left( \frac{\text{Depth}_{\text{Max}}}{V_v} \right)
\]  

(3.4)

The wings on this underwater glider has external control surfaces called flaps to aid in horizontal movement of the glider. The glider performance values shown in Table 3.1 are calculated using a flap deflection of 15° in the downward direction. The variables in the vehicle performance calculations that include factors concerning the flap deflection are the coefficients of drag and lift for the wings. Flap deflection is discussed more in detail in Chapter 4.

<table>
<thead>
<tr>
<th>Table 3.1: Glider Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Depth</td>
</tr>
<tr>
<td>Glide Angle</td>
</tr>
<tr>
<td>Velocity</td>
</tr>
<tr>
<td>Horizontal Velocity</td>
</tr>
<tr>
<td>Vertical Velocity</td>
</tr>
<tr>
<td>Time of Full Dive</td>
</tr>
<tr>
<td>Range of Full Dive</td>
</tr>
</tbody>
</table>

Figure 3.6 displays a graph of wing flap deflection angle verses the glide angle using Equation 3.1. For glide slopes similar to current gliders, the flap must be deflected 15-20°.
Figure 3.6: Flap Deflection vs. Glide Angle

Figure 3.7 displays the predicted sawtooth pattern of the glider dive profile down to 1000 meters. This dive profile does not account for environmental, buoyancy, or velocity changes of the glider.

The wing design and the fairing of the glider has a major impact on the glide slope, range, and time of the glide. The desired velocity of the glide, the density, and the areas and coefficients of the tail, wings, and body can determine the buoyancy needed for a steady glide. This buoyancy greatly impacts the vertical movement of the glide.
3.3 Buoyancy and Stability

A critical concept concerning the vertical movement of a glider is vehicle buoyancy. A vehicle with a positive buoyancy allows the vehicle to move upward, and a negative buoyancy allows the vehicle to move downward [14]. Archimedes’ Principle is the main concept underlying the buoyancy of underwater vehicles. When a vehicle is submerged in water, a buoyant force acts on the body vertically upward due to the pressure forces below the submerged body being greater than the pressure forces above. The buoyant force results in a value equal to the weight it displaces [15]. Finding the buoyant force is done by analyzing a free-
body diagram of the vehicle during an unaccelerated glide, using Equation 3.5.

\[
\begin{align*}
\sum F_x &= 0 \\
\sum F_y &= 0
\end{align*}
\] (3.5)

Summing up all forces on the glider, the buoyant force \( B \) equation is shown in Equation 3.6.

\[
B = \rho V^2 (A_W C_{LW} \cos \theta + A_W C_{DW} \sin \theta + A_B C_{DB} \sin \theta + A_T C_{DT} \sin \theta) \] (3.6)

From the buoyant force, the change in volume needed for the buoyancy engine for a full dive is calculated in Equation 3.7 using gravity \( g \) as 9.81 m/s (1.99 slugs/ft\(^3\)) [14]. The buoyancy change in volume \( \delta V_{ol} \) can change throughout a dive and the buoyant force must be balanced by the vertical components of lift and drag in order to obtain a steady glide.

\[
\delta V_{ol} = \frac{2B}{\rho g}
\] (3.7)

Figure 3.8 displays a basic concept of a buoyancy engine for the glider. This thesis does not cover the design and building of the buoyancy engine but presents an overview of concepts involved with the design of a buoyancy engine.

Using Figure 3.8 for the explanation of the buoyancy of an underwater glider, the dashed square shows the area of the vacuum. Using Equation 3.6 and Equation 3.7, this underwater glider has a change in volume of 1.82 liters (110.8 in\(^3\)). Using the dashed square as a control mass \( m \), the external inflatable bladder is external to the control mass. While descending, hydraulic fluid moves from external inflatable bladder which is at a high pressure to the internal
reservoir which is at a low pressure through a valve. Referring to Equation 3.8, the decrease in volume (Vol) creates an increase in density, causing negative buoyancy. While ascending, hydraulic fluid moves from internal accumulator to the external inflatable bladder by the pump. The increase in volume creates a decrease in density causing positive buoyancy. At this state seawater also flushes out of the vehicle, aiding in the ability to rise to the surface. For a neutral buoyancy, the vehicle must have a density equal to seawater.

\[ \rho = \frac{m}{Vol} \]  \hfill (3.8)

Vehicle trimming is critical for underwater vehicles. This process adds weight or foam in order to increase or decrease buoyancy of the vehicle to obtain neutral or slightly positive buoyancy. The fixed weight (W) of the vehicle in Equation 3.9 is the weight of the vehicle and the buoyant force is the result of Equation 3.6. Foam or weight is added in order to have equal weight and buoyant force. If the weight of the vehicle exceeds the buoyant force, foam is added to the vehicle to add buoyancy.

\[ W = B + F_{foam} - F_{weight} \]  \hfill (3.9)
Gliders are controlled through hydrostatics and manipulate hydrostatic balances in order to accomplish roll and pitch of the vehicle. Stability of the vehicle is a major factor for underwater vehicles. A stable vehicle has the center of gravity below the center of buoyancy. In this case, the weight of the vehicle creates a restoring moment to add stability to the vehicle. Roll and pitch on the glider is accomplished by increasing the lift on the wings. The following chapter details the conceptual design of glider wings and how lift is achieved.
Chapter 4

Wings

Underwater gliders use wings to generate a hydrodynamic lift force to propel the vehicle forward, provide control, and provide stability. The design of the airfoil section is a major component and predicts how the vehicle glides during a dive and steady flight dynamics are used to predict forces created from the airfoil. The following chapter defines wing geometry and forces critical to airfoil and wing design.

4.1 Airfoil Nomenclature

Airfoils are classified as either symmetrical or cambered airfoils. A cambered airfoil has a distance between the mean camber line and the chord line. Looking at the airfoil nomenclature by Anderson the mean camber line is the line formed halfway between the upper and lower surfaces cutting the airfoil in half and the chord line is the straight line connecting the leading and trailing edges. A symmetrical airfoil has zero camber resulting in the mean chamber line equaling the chord line [16]. An external control surface flap placed on a symmetrical
airfoil when deflected upward or downward increases the camber of the airfoil.

Fluid flows from the leading edge to the trailing edge of the airfoil, creating streamlines. In laminar flow, streamlines are smooth, allowing fluid to move smoothly along a surface [16]. In turbulent flow, streamlines break up and fluid moves randomly. Laminar flow is necessary around an airfoil in order to decrease drag on the vehicle. A standard airfoil section has maximum thickness close to the leading edge of the airfoil. This results in the minimum pressure to be at the point of maximum thickness and increasing pressure to the trailing edge of the airfoil. In this case, a turbulent boundary layer forms due to the increasing pressure distribution [16]. A laminar airfoil section has maximum thickness near the middle and trailing edge of the airfoil. In this case, a decreasing pressure distribution is formed where laminar flow can develop and become turbulent toward the trailing edge [16].
4.2 Lift and Drag

Lift occurs on a wing when pressure under the airfoil is greater than pressure on top of the airfoil. As shown in Figure 4.2, the lift force is perpendicular to the velocity flow direction. Drag, the component parallel to the velocity flow direction is caused by the imbalance of pressure forces on the rearward surface and the forward surface of the airfoil [16]. The angle of attack ($\alpha$) is the angle between the fluid velocity vector and the chord line and is used to determine values of lift and drag. As flow separates around an airfoil, lift decreases and drag increases. Drag also increases when air flows around the tips of a wing from the high to low pressure. This air flow creates a circulatory motion that occurs downstream of the wings. This circulatory motion, also called a vortex, creates a velocity component in the downward direction at the wing, creating a downwash [16].

Figure 4.2: Normal and Axial Force on an Airfoil [16]
Wing equations for lift and drag forces on a vehicle use coefficients of lift and drag for finite wings, which are 3-dimensional wings stretching in the z-direction a distance known as the wing span. Finite wing coefficients are displayed as $C_L$ and $C_D$. Equations to calculate the finite wing coefficients use infinite wing coefficients. Infinite wings stretch to infinity in the z-direction and are 2-dimensional, allowing flow only in the x- and y-directions. Infinite wing coefficients are displayed as $c_l$ and $c_d$. These coefficients are taken from experimental data for many airfoil sections in wind tunnels compiled over the years from the National Advisory Committee for Aeronautics (NACA) and the National Aeronautics and Space Administration (NASA) [16]. In the following wing design, software called Javafoil is used to obtain infinite wing coefficients. Javafoil estimates lift and drag infinite coefficients of the airfoil based on the Panel Method for inviscid incompressible fluids [17, 18].

Figure 4.3: The Effect of a Cambered and Uncambered Airfoil [16]
Figure 4.3 displays an example of a lift curve for a cambered airfoil and uncambered airfoil. For cambered airfoils, the airfoils have zero lift at a negative angle of attack. A symmetrical airfoil, also known as an uncambered airfoil, have zero lift at a zero angle of attack. Figure 4.3 also displays the coefficient of lift of a finite wing ($C_L$) varying linearly with $\alpha$ until the wing is stalled. Stall happens when flow separation occurs above the airfoil and lift begins to rapidly decrease [13].

### 4.3 Flaps

A plain flap deflected on an airfoil is used to add additional lift, increase camber, increase the angle of attack, and decrease the stalling angle of attack [16]. Figure 4.4 illustrates how lift is effected by a flap deflection. The reduction of the stall angle is done by increasing the pressure drop over the top of the airfoil during deflection, promoting flow separation [13]. From the figure, it is shown that the lift slope remains unchanged and the deflected wing has the zero angle of attack shifted to a more negative value [16].

As the wing remains stable, the flap is moved up or down to control the pitch and roll of the glider during ascent and descent. Lift is increased as the flap moves upward on descent and lift is increased as the flap moves downward on ascent.
Figure 4.4: Effect of Flaps [16]
4.4 Wing Design

The following equations for the coefficient of lift, coefficient of drag, lift force, and drag force are from textbooks on aerospace engineering that cover the flight dynamics and the airfoil of an airplane. The following equations factor in major variables, such as kinematic viscosity and density, which distinguish between air and water. Kinematic viscosity and vehicle traveling velocity both factor into the Reynolds Number, a major value affecting how fluid flows. In reality, airplanes do not travel at the velocity of 0.25 m/s experienced by gliders, and because of this, airplanes experience a much higher Reynolds Number. Just like airplane models and airplane airfoils, any value of theoretical data must be tested. It is seen in [8] and [3] that each glider has gone through multiple tests in wave tunnels, wave tanks, and lakes. The differences between the theoretical estimates and the experimental data are significant.

According to Equation 4.1, the wings and tail of the glider operate at a low Reynolds number (Re) in the laminar flow region due to the slow vehicle velocity (V), where L is the characteristic length and $\gamma$ is the kinematic viscosity of the fluid [19].

$$ Re = \frac{VL}{\gamma} \quad (4.1) $$

The wing Reynolds number is 66,558, which is closer to 40,000 and can be directed toward the graph on the left of Figure 4.5. This shows lift and drag characteristics of a cambered, flat plate, and airfoil wing sections at low turbulence and low Reynolds number. In order for an airfoil to perform best at a low Reynolds number, the thickness ratio must be low and the chamber must be high. At a higher Reynolds number, a thicker airfoil gives better performance.
Since underwater gliders have potential to experience turbulent flow, the thin symmetrical airfoil was chosen for better performance [19]. The flap on the wing provides added chamber for added performance.

![Figure 4.5: Low Reynolds Number on Wing Design [19]](image)

The wing’s aspect ratio (AR) affects the performance of the glide. The aspect ratio is calculated using Equation 4.2 where b is the wing span and $A_W$ is the wing planform area [13]. Figure 4.6 displays a plot showing the effect of the aspect ratio on lift. A wing with a higher wing span is less affected by the tip vortex and has a higher aspect ratio. High aspect ratio wings do not experience as much of a decrease in lift and increase in drag due to the tip effects as a low aspect wing of the same wing area. A high aspect wing has less drag for a given lift. It can be concluded from the graph that the stalling angle changes as the aspect ratio changes. Consequently, the lower aspect ratio lift curves stall at a higher angle of attack ($\alpha$) [13].
The Oswald efficiency \( e \) is another variable factored into lift coefficient calculations and accounts for the extra drag due to non elliptical lift distribution and the flow separation on the airfoil. The maximum Oswald span efficiency that can occur is 0.98 and typically occurs between 0.7 and 0.85 \([13]\). Equation 4.3 calculates the Oswald efficiency for glider wings swept 35° to prevent underwater entanglement \([13]\).

\[
e = 4.61 \left( 1 - 0.045A_W^{0.68} \right) \left( \cos \wedge_{LE} \right)^{0.15} - 3.1
\]  
(\(\wedge_{LE} > 30\))
4.4.1 Coefficient of Lift for Finite Wings

The wing coefficient of lift ($C_{LW}$) for finite wings in Equation 4.4 takes into account the finite wing lift slope coefficient ($a$) and the angle of attack ($\alpha$) in degrees [19].

$$C_{LW} = a (\alpha - \alpha_0)$$  \hspace{1cm} (4.4)

The lift slope coefficient of an infinite wing ($a_0$) is calculated using Equation 4.5 using the infinite coefficient of lift ($c_l$) [19] and converted for finite wings in Equation 4.6 using the Oswald Efficiency and Aspect Ratio [19].

$$a_0 = \frac{c_l}{\alpha - \alpha_0}$$  \hspace{1cm} (4.5)

$$a = \frac{a_0}{1 + \frac{57.3a_0}{\pi e AR}}$$  \hspace{1cm} (4.6)

The calculation of the lift slope coefficient in Equation 4.6 is for a wing that extends from wing tip through the fairing to the center of the vehicle. In the case of this underwater glider, a portion of the wing on each side is covered by the fairing. Equation 4.7 accounts for the fairing diameter and is named in this thesis as the Raymer Method taking into account the wing planform area exposed ($S_{Exposed}$) and wing planform area reference ($S_{Ref}$) [13]. Since gliders travel at subsonic speeds, the Mach Number ($M_N$) is less than 1.
\[ a_r = \frac{2\pi AR}{2 + \sqrt{4 + \frac{AR\beta^2}{\eta^2}} \left(1 + \frac{\tan^2\Delta LE}{\beta^2}\right)} \left(\frac{S_{\text{Exposed}}}{S_{\text{Ref}}}\right) F \]  \hspace{0.5cm} (4.7)

\[ \beta^2 = 1 - M_N^2 \]

\[ \eta = \frac{a_0}{2\pi/\beta} \]

\[ F = 1.07 \left(1 + \frac{d}{b}\right)^2 \]

Wing coefficient of lift is similar to Equation 4.4 but takes into account the new lift slope coefficient \(a_r\) [19].

\[ C_{LW} = a_r (\alpha - \alpha_0) \]  \hspace{0.5cm} (4.8)

### 4.4.2 Coefficient of Drag for Finite Wings and Tail

According to Equation 4.9, the coefficient of drag \(c_d\) for the infinite wing obtained from Javafoil includes skin friction drag \(c_{df}\), pressure drag \(c_{dp}\), and wave drag coefficients \(c_{dw}\) [13].

\[ c_d = c_{df} + c_{dp} + c_{dw} \]  \hspace{0.5cm} (4.9)

The coefficient of drag for a finite wing \(C_{DW}\) shown in Equation 4.10 takes into account the Oswald efficiency and aspect ratio [19].

\[ C_{DW} = c_d + \frac{C_{LW}^2}{\pi e AR} \]  \hspace{0.5cm} (4.10)

The tail on an underwater glider is mainly used to provide stability, generates no lift. Because of this, the coefficient of drag for the tail \(C_{DT}\) in the infinite
case is the same as the finite case as seen in Equation 4.11. The coefficient of drag is taken directly from Javafoil and measured at zero angle of attack.

\[ C_{DT} = c_d \]  

(4.11)

### 4.4.3 Lift Force and Drag Force

Finally, calculating the total forces of lift and drag on the vehicle can be done using the lift force \( F_L \) in Equation 4.12 and the drag force \( F_D \) in Equation 4.13 [19]. To find the drag force on the tail, \( C_{DW} \) is replaced with \( C_{DT} \) in Equation 4.13.

\[ F_L = \frac{1}{2} \rho V^2 A_c C_{LW} \]  

(4.12)

\[ F_D = \frac{1}{2} \rho V^2 A_c C_{DW} \]  

(4.13)

Using the coefficients of lift and drag for the tail and wing to calculate the force of drag and lift on the vehicle factored into the selection of the airfoil. Trail and error was performed when deciding on the type of airfoil that gives the better performance for this glider. Coefficients of lift and drag and forces of lift and drag were calculated and divided to calculate the lift to drag ratio. The lift to drag ratio for wings chosen for this glider is 17.6, calculated from dividing the coefficient of lift over the coefficient of drag at a flap deflection of 15°. A higher lift to drag ratio is ideal. It is concluded from calculations that a symmetrical thin airfoil with a flap to add camber is ideal for underwater gliders operating at a low Reynolds Number.
4.5 Glider Wings

The airfoil chosen for this glider is a modified NACA 65-010 airfoil with the location of greatest thickness at 50% of the chord as seen in Figure 4.7. The wing is fixed on the glider with a flap at the trailing edge to provide lift. The wing is tapered and swept 35°.

![Figure 4.7: Airfoil](image)

The wing is analyzed for infinite wing coefficients in Javafoil. The final drag and lift coefficients use the Raymer Method for analysis, taking into account the fairing diameter. Table 4.1 and Table 4.2 display calculated values of lift and drag. The calculations predict very small drag values for the wing and tail.

<table>
<thead>
<tr>
<th>Table 4.1: Wing Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design Velocity</td>
</tr>
<tr>
<td>Reynolds Number of Wings</td>
</tr>
<tr>
<td>Coefficient of Lift of Infinite Wing (JavaFoil)</td>
</tr>
<tr>
<td>Coefficient of Drag of Infinite Wing (JavaFoil)</td>
</tr>
<tr>
<td>Wing Tip Chord with Flap</td>
</tr>
<tr>
<td>Wing Root Chord with Flap</td>
</tr>
<tr>
<td>Wing Span (tip to tip)</td>
</tr>
<tr>
<td>Wing Aspect Ratio</td>
</tr>
<tr>
<td>Coefficient of Drag on Finite Wings ($C_D$)</td>
</tr>
<tr>
<td>Coefficient of Lift on Finite Wing ($C_L$)</td>
</tr>
<tr>
<td>Force of Drag per wing</td>
</tr>
<tr>
<td>Force of Lift per wing</td>
</tr>
</tbody>
</table>
Table 4.2: Tail Results

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design Velocity</td>
<td>0.40 m/s</td>
</tr>
<tr>
<td>Reynolds Number of Tail</td>
<td>39,402</td>
</tr>
<tr>
<td>Tail Tip Chord</td>
<td>6.35 cm</td>
</tr>
<tr>
<td>Tail Root Chord</td>
<td>12.7 cm</td>
</tr>
<tr>
<td>Coefficient of Drag on Finite Tail ($C_D$)</td>
<td>0.0149</td>
</tr>
<tr>
<td>Drag force on the Tail</td>
<td>0.0178 N</td>
</tr>
</tbody>
</table>

4.5.1 Construction

The wing mold was completed by the Florida Institute of Technology machine shop CNC machine using 10-12 lb/in Polyurethane foam. The completed mold, shown in Figure 4.8(a), was sprayed with gelcoat and sanded to provide a smooth finish for fiberglassing. After the foam was sanded, it was covered with 2 layers of fiberglass and Vinyl Ester resin. The flap was fiberglassed with a fiberglass cloth to provide bending flexibility. The completed lay and the wings out of the mold are shown in Figure 4.8(c) and Figure 4.8(d).

Once out of the mold, each wing half was trimmed, sanded, and fitted to the other half of the wing. Both left and right wings were cut and fitted similarly to have equal flaps and provide equal lift. The top and bottom half of each wing were glued with an epoxy adhesive, and the epoxied areas were covered by fiberglass chop to provide a strong and smooth surface. The wings and tail contain holes to provide water flow into the wing and tail to avoid air pockets. Brass threaded inserts in Figure 4.14 are placed in fiberglass to provide a structure at the mounting point of each wing and tail. Stainless steel bolts screw into the brass threaded inserts to mount the wing and tail. Figure 4.10 and Figure 4.11 show the completed CAD drawing and actual wing and tail.
Figure 4.8: Building the Glider Wings

(a) Completed CNC
(b) Sanded
(c) Wing Lay with Fiberglass
(d) Wings Out of Mold

Figure 4.9: Brass Threaded Insert
4.5.2 Flap Control

For flap control, a servo motor is connected to a control rod and a control horn to deflect the wing flap into a positive and negative 20° which is shown in Figure 9.6(b). The servo is sent a pulse width modulated signal of a certain frequency and duty cycle to rotate the servo enough to deflect the wing, discussed in detail in Chapter 7.7.

The positioning of the control horn, the length of the servo arm, and the length of the control rod has a major influence on the bending capability of the
flap. From testing the control horn mechanism, these points were determined.

- The location of the control horn must be placed directly above the hinge point
- The control rod must be as close as possible to the hinge point on the flap side, or at the lowest setting of the control horn
- The control rod must be connected to the very top setting of the servo arm

For the flaps on the wings, the control horn was placed not directly above the hinge point of the wing, but closer to the end of the flap. This decreases the torque needed by the servo motor to deflect the flap. The control horn is drilled into the flap, and then epoxied to the flap to provide adequate strength.

### 4.5.3 Wing Mounting

The mounting of the wing to the glider is shown in Figure 4.13. The servo motor mounts to a piece of plastic, which is attached to the wing, allowing for
removal of the wing control. Bulkheads are epoxied to the sides of the fairing which holes for bolts for wing attachment which allows easy removal of the wings. This mounting technique also provides movement forward or aft to test the wing positioning for future glider tests.

![Figure 4.13: Wing Mount](image)

**4.5.4 Tail Mounting**

The mounting of the tail to the glider consists of placing a small thick bulkhead in the tail section. This fiberglass bulkhead is epoxied and glassed to the bottom half of the hull. This bulkhead contains a threaded insert. A piece of High Density Polyethylene (HDPE) plastic is cut to attached to the bottom of the tail. This plastic is screwed into the bulkhead to provide stability.

Figure 4.15 displays the vehicle with the tail and wings attached. The glider fairing was cut out to fit the wings and provide adequate wing flap movement.
Figure 4.14: Tail Mount

Figure 4.15: Glider with Appendages
Chapter 5

Fairing

The primary function of the fairing of an underwater glider is to produce a smooth outline for the glider, reduce drag, improve performance, and provide a structure for internal components to mount to. Fluid flow and drag characteristics discussed in Chapter 4 also apply to the fairing.

The Reynolds Number (Re) in Equation 5.1 characterizes the flow around the fairing and is used to determine the coefficient of frictional drag \(C_{FB}\) shown in Equation 5.2. Using Equation 5.3 and the length (L) and diameter (D) of the body, the coefficient of drag \(C_{DB}\) is calculated for a streamlined body [8, 20].

\[
Re = \frac{vL}{\gamma} \quad \quad (5.1)
\]

\[
C_{FB} = \frac{0.075}{(\log_{10} Re - 2)^2} \quad \quad (5.2)
\]

\[
C_{DB} = C_{FB} \left[ 3 \left( \frac{L}{D} \right) + 4.5 \sqrt{\left( \frac{L}{D} \right)} + 21 \left( \frac{D}{L} \right)^2 \right] \quad \quad (5.3)
\]
The drag force ($F_D$) of the fairing is calculated using Equation 5.4 taking into account the traveling velocity ($v$), water density ($\rho$), and fairing cross-sectional area ($A_C$) [19].

$$F_D = \frac{1}{2} \rho v^2 A_C C_{DB}$$  

(5.4)

Table 5.1 shows the calculated values using the equations for the glider fairing in Figure 5.1. It can be concluded that fairing has a high impact on the drag of the overall vehicle.

<table>
<thead>
<tr>
<th>Design Velocity (v)</th>
<th>0.40 m/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reynolds Number of the Body (Re)</td>
<td>764189.51</td>
</tr>
<tr>
<td>Fairing Length Overall (L)</td>
<td>1.97 m</td>
</tr>
<tr>
<td>Fairing Diameter (D)</td>
<td>0.35 m</td>
</tr>
<tr>
<td>Aspect Ratio (AR)</td>
<td>5.7</td>
</tr>
<tr>
<td>Coefficient of Drag on the Fairing ($C_{DB}$)</td>
<td>0.54</td>
</tr>
<tr>
<td>Force of Drag on the Fairing ($F_D$)</td>
<td>4.23 N</td>
</tr>
<tr>
<td>Total Drag Force for the Glider</td>
<td>4.54 N</td>
</tr>
</tbody>
</table>

Construction of the fairing began on October 18, 2009. The mold was previously designed and implemented by Alan Shaw, who contributed to Florida Institute of Technology. Three bulkheads were placed inside to provide stability,
support for the internal components, and mountings for the pressure hull. The bottom of the hull was constructed with 3 layers of fiberglass mat and the top of the hull was constructed with 2 layers of fiberglass mat. The bulkheads are supported by a higher strength fiberglass mat. Figure 5.2 displays the construction of the hull and the final product with bulkheads supporting the pressure housing.

(a) Applying Resin
(b) Laying the fiberglass mat
(c) Reinforcing the tail
(d) Bulkheads

Figure 5.2: Building the Glider Fairing
Chapter 6

Pressure Housing

Pressure hulls are necessary for underwater gliders to store the computer and electronics in a dry place away from effects of pressure. External hydrostatic pressure at ocean depths can cause shell yielding, shell buckling, general instability, and compressibility failures for axi-symmetric ring-stiffened cylinders. This chapter explains all failure modes with equations to determine the maximum operating depth of the pressure housing.

6.1 Yielding

Shell yielding is the point where the material deforms permanently, also known as deforming plastically. If the pressure hull undergoes shell yielding, yielding occurs at fibers along the outer surface of the shell between the stiffeners and the inner surface of the shell at the stiffeners. Yield failure creates folds of the cylinder between the stiffeners, undergoing a large deformation. Equation 6.1 calculates the axi-symmetric yield pressure ($P_y$) of the pressure housing subject to external pressure where $\sigma_y$ is the yield strength, $t$ is the thickness, $R$ is
the shell radius, $b_f$ is the width of a structural stiffener, $A_f$ is the area of the stiffener, $\nu$ is the Poisson’s ratio of the material, and $L_f$ is the length between the structural stiffeners [21].

$$P_y = \frac{\sigma_y r}{1 + H \left( \frac{0.85-B}{1+\beta} \right)}$$  \hspace{1cm} (6.1)

where,

$$B = \frac{b_t}{A_f + b_f t}$$

$$\beta = \frac{11N}{\sqrt{\frac{50t}{R}}} \left( \frac{t^2}{A_f + b_f t} \right)$$

$$N = \frac{\cosh \theta - \cos \theta}{\sinh \theta + \sin \theta}$$

$$\theta = 10 \left[ 12 \left( 1 - \nu^2 \right) \right]^{\frac{1}{2}} \left( \frac{L_f}{2R} \right) \left( \frac{50t}{R} \right)^{-\frac{1}{2}}$$

$$H \approx -\frac{3 \sinh \left( \frac{\theta}{2} \right) \cos \left( \frac{\theta}{2} \right) + \cosh \left( \frac{\theta}{2} \right) \sin \left( \frac{\theta}{2} \right)}{\sinh \theta + \sin \theta}$$

### 6.2 Buckling

Buckling is the failure of a structure in response to compressive stress. In a stiffened cylinder, the frames have a greater resistance to buckling. The pressure creates a buckle or deformation in the cylinder, giving the cylinder the appearance of dimples between stiffeners. Equation 6.2 calculates the buckling ($P_b$) of a stiffened pressure vessel where $E$ is the modulus of elasticity of the material [21].

$$P_b = \frac{2.42E}{(1 - \nu^2)^{3/4}} \left[ \frac{L_f}{2R} - 0.45 \sqrt{\frac{t}{2R}} \right]^{5/2}$$  \hspace{1cm} (6.2)
6.3 General Instability

General Instability is caused by a buckling combination of the frame and shell. Instability creates dented surfaces on the pressure housing and appear less than the dimples shown in a buckling failure [21]. The pressure at which the instability failure occurs is calculated using Equation 6.3 where \( n \) is the integer number resulting in the lowest value of \( P_{cr} \) and in this case \( n=2 \), \( A \) is the cross sectional area of ring reinforcement, \( \xi \) is the distance from the surface of the hull to the neutral axis of ring reinforcement, and \( I \) is the moment of inertia of the stiffener [21, 22].

\[
P_{cr} = \frac{Et}{R} \left( n^2 + \frac{m^2}{2} - 1 \right) \left( n^2 + m^2 \right)^2 + \frac{(n^2 - 1)}{R^3 L_f} E I_l
\]

\[
I_l = A \left( \frac{\xi + \frac{L}{2}}{1 + \frac{A}{L_f t}} \right) + I + \frac{L_f t^3}{12}
\]

\[
m = \frac{(\pi R)}{L}
\]

6.4 Compressibility

Compressibility is an important factor in underwater pressure hull design due to the need of a vehicle to maintain neutral buoyancy. If the pressure hull is more compressible than seawater, the vehicle ballast must be discharged during descent to maintain neutral buoyancy. If the pressure hull is less compressible than seawater, ballast must be added during descent. Presented here is a simplified equation for the compressibility of spherical pressure hulls. Equation 6.5 returns the cubic inch change in volume of a cylindrical section between adjacent stiffeners per foot of sea water depth [21]. This equation must be multiplied by the number of sections between stiffeners and the maximum depth in meters in...
order to get the volume lost in cubic inches due to cylinder compression. In this case, there are 3 sections between the stiffeners and the maximum design depth is 1000 m (3280 ft). This value is multiplied by a factor of 16.39 to convert cubic inches to cubic centimeters.

\[
C_c = \frac{64 \frac{h}{ft^2}}{144 \frac{m^2}{ft^2}} \frac{\pi R^3 L_f (2 - \nu)^2}{2Et} \left[ 1 + \frac{(1 - \nu^2)}{(2 - \nu)^2} - \frac{1}{\frac{L_f}{A_f} + \frac{\theta}{2N}} \right] \tag{6.5}
\]

where,

\[
\theta = \left[ 3 (1 - \nu^2) \right]^{\frac{1}{2}} \frac{L}{(Rt)^{\frac{1}{2}}}
\]

\[
N = \frac{\cosh \theta - \cos \theta}{\sinh \theta + \sin \theta}
\]

### 6.5 End Plate Stress

The pressure housing for this glider contains flat end plates, which are bolted to the cylinder around the outside edge of the end plate. The flat end plate must be able to survive flexural pressure, and have an end plate thickness greater than the cylinder. End plate stress equations are calculated for the area not constrained by fasteners and assumes there is a constant pressure along the end plate. The bending stress for flat circular plates of constant thickness is shown in Equation 6.6 where \( M \) is the bending moment and \( t \) is the end plate thickness [23].

\[
\sigma = \frac{6M}{t^2} \tag{6.6}
\]

Equation 6.7 and Equation 6.8 calculate the moment at the center \( (M_c) \) and moment at the reaction \( (M_{ra}) \) for uniformly distributed pressure with fixed sup-
ports where \( q \) is the maximum pressure at the design depth and \( r \) is the radius of the diameter of the end plate that is not constrained. These moment equations are put into Equation 6.6 to calculate the stress at the center(\( \sigma_c \)) shown in Equation 6.9 and the stress at the reaction(\( \sigma_{ra} \)) shown in Equation 6.10 [23].

\[
M_c = \frac{-qr^2 (1 + \nu)}{16} \quad (6.7)
\]

\[
M_{ra} = \frac{-qr^2}{8} \quad (6.8)
\]

\[
\sigma_c = \frac{-6qr^2 (1 + \nu)}{16t^2} \quad (6.9)
\]

\[
\sigma_{ra} = \frac{-6qr^2}{8t^2} \quad (6.10)
\]

The maximum stress calculated is used in the design. The equations were rearranged in order to solve for the minimum thickness of the end plate in order to withstand the pressure at depth. A factor of safety of two is applied to all strength calculations. Equation 6.12 calculates the end plate deflection for the end plate thickness [23]

\[
t_{min} = \max \left( \sqrt{\frac{-3qr^2 (1 + \nu)}{8\sigma_y}}, \sqrt{\frac{-3q}{4\sigma_y}} \right) \quad (6.11)
\]

\[
y_c = \frac{-6qr^2 (1 - \nu^2)}{32Et^3} \quad (6.12)
\]
6.6 Design

The pressure housing constructed for the glider is an Aluminum 6061-T6 cylinder of 0.25 inch thickness stiffened by aluminum stiffeners approximately at every six inches. Figure 6.1 displays the computer-aided design (CAD) model of the pressure hull and the constructed hull.

![Figure 6.1: Pressure Hull CAD Drawing and Constructed Hull](image)

Based on the results found in Table 6.1, the maximum depth for the pressure hull is 3,801 m (12,468 ft.) with an end plate thickness of 0.75 in. The pressure hull is predicted to fail to yield stress before instability and buckling. The maximum operating pressure in Table 6.1 is the minimum pressure acquired from all failure modes.

The pressure hull end plates were previously designed with a 0.25 in. thickness, as shown in Figure 6.1. End plate stress calculations presented in the design were calculated after the manufacturing of the pressure housing.

At both ends of the pressure housing, an o-ring is placed to create a seal to prevent leaking. The o-ring is designed according to the Parker O-Ring Handbook. The specifications are in Table 6.2.

The end plates are bolted down with Aluminum bolts on both sides of the
Table 6.1: Pressure Hull Design Data

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aluminum 6061-T6 Modulus of Elasticity</td>
<td>68.9 GPa</td>
</tr>
<tr>
<td>Aluminum 6061-T6 Yield Strength</td>
<td>275 MPa</td>
</tr>
<tr>
<td>Cylinder thickness</td>
<td>0.635 cm</td>
</tr>
<tr>
<td>Cylinder mean diameter</td>
<td>16.19 cm</td>
</tr>
<tr>
<td>Cylinder length</td>
<td>50.8 cm</td>
</tr>
<tr>
<td>Length between frames</td>
<td>16.93 cm</td>
</tr>
<tr>
<td>Frame thickness</td>
<td>1.27 cm</td>
</tr>
<tr>
<td>End plate thickness</td>
<td>1.91 cm</td>
</tr>
<tr>
<td>Max Operating Pressure</td>
<td>76.7 MPa</td>
</tr>
<tr>
<td>Max Operating Depth</td>
<td>7600.77 m</td>
</tr>
<tr>
<td>Rated Depth (Factor of Safety=2)</td>
<td>3801.0 m</td>
</tr>
<tr>
<td>Volume Lost to Compressibility (1000 m depth)</td>
<td>38.67 cc</td>
</tr>
</tbody>
</table>

Table 6.2: Dimensions for Park O-Ring 2-261

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>O-Ring Inner Diameter</td>
<td>17.124 ± 0.102 cm</td>
</tr>
<tr>
<td>Cross Section (W)</td>
<td>0.353 ± 0.010 cm</td>
</tr>
<tr>
<td>Groove ID</td>
<td>17.104 cm</td>
</tr>
<tr>
<td>Groove Width Max (G)</td>
<td>0.475 cm</td>
</tr>
<tr>
<td>Groove Width Min (G)</td>
<td>0.450 cm</td>
</tr>
<tr>
<td>Gland Depth Max (L)</td>
<td>0.272 cm</td>
</tr>
<tr>
<td>Gland Depth Min (L)</td>
<td>0.257 cm</td>
</tr>
</tbody>
</table>

A vacuum port on the pressure hull is used to apply a vacuum to the housing before deployment and a pressure relief valve is used to open the housing after deployment. Anodizing and other chemical treatments are needed in order to increase corrosion resistance of the pressure hull if the vehicle will be in the water for a long time.
Chapter 7

Gumstix Overo Fire

The computer and electronics of this underwater glider are mounted to the High Density Polyethylene (HDPE) holder shown in Figure 7.1.

Figure 7.1: Electronics Holder
Table 7.1 is an overview of all the sensors on board this glider that are integrated into the central computer. These sensors are discussed more in detail in this chapter.

Table 7.1: Overview of Sensors

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Voltage</th>
<th>Current</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inertial Measurement Unit</td>
<td>3.4-10 V</td>
<td>24 mA</td>
<td>Serial</td>
</tr>
<tr>
<td>Global Positioning System</td>
<td>3.3-5 V</td>
<td>65 mA</td>
<td>Serial</td>
</tr>
<tr>
<td>Compass</td>
<td>3.3-5 V</td>
<td>25 mA</td>
<td>Serial</td>
</tr>
<tr>
<td>Servo Motor</td>
<td>4.8-6.0 V</td>
<td>600 mA (stalled)</td>
<td>PWM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7.4 mA (stopped)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>160 mA (running-no load)</td>
<td></td>
</tr>
</tbody>
</table>

7.1 Python Programming Language

Python is the general purpose programming language used on the glider and is already built into the central computer. The multiple libraries for Python eases programming and speeds up sensor integration [24]. This portable language is fast and very powerful with readable syntax that boosts developer productivity. The library Pyserial is a module in Python that accesses the serial port to gather data. With the use of this module a data string can be read, split, and sent through the serial port easier than other programming languages.
7.2 Gumstix Overo Fire Computer and Tobi Expansion Board

The Gumstix Overo Fire is the central computer on the glider [25]. This computer is referred as the Overo Fire in the remaining text. The Overo Fire has a Texas Instruments OMAP 3530 Applications Processor [26] with a 720 MHz clock, 256MB of RAM, 256MB Flash, 802.11 b/g wireless communications, bluetooth communications, and microSD slot. The Overo Fire has expandability and is powered by a Tobi Expansion board [25]. The Overo Fire is connected to the Tobi Expansion board via two 70-pin connectors. Figure 7.2 shows the Overo Fire computer connected to the Tobi Expansion board.

Figure 7.2: Gumstix Overo Fire and Tobi Expansion Board
These 70-pin connectors provide the ability to have a 40-pin header on the Tobi Expansion board allowing for:

- I²C Port
- Six PWM lines
- Six A/D input lines
- SPI Bus
- Two two-wire serial ports
- One 1-wire port
- USB Serial Console
- Stereo Audio In and Out
- USB OTG mini-AB
- USB host standard A

Table 7.2 displays the pin number and what signal is connected to each pin. Since signals are at 1.8 volt logic, a logic converter from Sparkfun Electronics is used to convert logic from High (5 volts and 3.3 volts) to low (1.8 volts) or low (1.8 volts) to high (5 volts and 3.3 volts) [27].

The logic level converter is shown in Figure 7.3. This board is powered by a high voltage and a low voltage. The converter contains a BSS138, which is an N-Channel logic level enhancement mode field effect transistor [27]. These transistors are reliable and fast-switching to provide the conversion of the signal. Any digital signal at a high voltage is inputted through the board at the high voltage side, and it is converted to a low voltage signal and outputted on the low voltage side.
Table 7.2: Overo Fire 40-Pin Header on Tobi Expansion Board

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Sensor</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>GPIO151_RXD1</td>
<td>TXI of Logic Converter, TX of GPS</td>
</tr>
<tr>
<td>10</td>
<td>GPIO148_TXD1</td>
<td>RXO of Logic Converter, RX of GPS</td>
</tr>
<tr>
<td>15</td>
<td>GND</td>
<td>Ground of Logic Converter for GPS and Compass</td>
</tr>
<tr>
<td>16</td>
<td>VCC_1.8</td>
<td>Low Voltage of Logic Converter for GPS and Compass</td>
</tr>
<tr>
<td>21</td>
<td>GPIO165_IR_RXD3</td>
<td>TXI of Logic Converter, TX of Compass</td>
</tr>
<tr>
<td>22</td>
<td>GPIO166_IR_TXD3</td>
<td>RXO of Logic Converter, RX of Compass</td>
</tr>
<tr>
<td>25</td>
<td>GND</td>
<td>Ground of Logic Converter for Servo Motors</td>
</tr>
<tr>
<td>26</td>
<td>VCC_1.8</td>
<td>Low Voltage of Logic Converter for Servo Motors</td>
</tr>
<tr>
<td>27</td>
<td>GPIO146_PWM11</td>
<td>Wing Control Left Wing</td>
</tr>
<tr>
<td>28</td>
<td>GPIO145_PWM10</td>
<td>Wing Control Right Wing</td>
</tr>
<tr>
<td>29</td>
<td>GPIO147</td>
<td>GPIO Abort Signal</td>
</tr>
</tbody>
</table>

Figure 7.3: Logic Level Converter
7.3 Global Positioning System (GPS)

The PMB-688 Global Positioning System (GPS) module in Figure 7.4 purchased from Parallax Inc, connects serially to the Overo Fire through ttyS2 at 4800 bps [28]. A python code using the pyserial library parses the GPS string starting with $GGRMC into coordinates. Table 7.3 shows an example string and the information obtained from parsing the GPS string.

![GPS Module](image)

Figure 7.4: GPS

Table 7.3: RMC-Recommended Minimum Data [29]

<table>
<thead>
<tr>
<th>Name</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message ID</td>
<td>$GPRMC</td>
<td>RMC protocol header</td>
</tr>
<tr>
<td>UTC Time</td>
<td>161229.487</td>
<td>hhmmss.sss</td>
</tr>
<tr>
<td>Status</td>
<td>A</td>
<td>A=data valid, V=data not valid</td>
</tr>
<tr>
<td>Latitude</td>
<td>3723.2475</td>
<td>ddmm.mmmm</td>
</tr>
<tr>
<td>N/S Indicator</td>
<td>N</td>
<td>N=north or S=south</td>
</tr>
<tr>
<td>Longitude</td>
<td>12158.3416</td>
<td>dddmm.mmmm</td>
</tr>
<tr>
<td>E/W Indicator</td>
<td>W</td>
<td>E=east or W=west</td>
</tr>
<tr>
<td>Speed Over Ground (knots)</td>
<td>0.13</td>
<td>E=East or W=west</td>
</tr>
<tr>
<td>Course Over Ground (deg)</td>
<td>309.62</td>
<td>True</td>
</tr>
<tr>
<td>Date</td>
<td>120598</td>
<td>dddmmyy</td>
</tr>
<tr>
<td>Magnetic Variation (deg)</td>
<td></td>
<td>E=east or W=west</td>
</tr>
</tbody>
</table>
7.4 Inertial Measurement Unit (IMU)

The movement of the underwater glider can be analyzed by measuring six degrees of freedom. Figure 7.5 displays the movement and rotation about the three axes relative to the underwater glider.

The Atomic 6 Degrees of Freedom Inertial Measurement Unit (IMU) from Sparkfun Electronics contains one Freescale MMA7260Q TM triple-axis accelerometer that measures acceleration along the X, Y, and Z axes, and three Microelectronics LISY300AL TM single-axis gyros that measures the rate of rotation around the X, Y, and Z axes. The data acquired from the accelerometer is used to measure vehicle movement on each axis by taking the double integral over time [30].

The IMU displayed in Figure 7.6 connects to the Overo Fire through an FTDI serial-to-usb converter using /dev/ttyUSB0 at 115200 bps. The USB is connected to the Overo Fire via a powered USB hub. The IMU is horizontally placed in front of the pressure housing aligned with the centerline. The output format of the data in ASCII is displayed in Table 7.4. The IMU is configured
with an ASCII output, 10kHz sampling rate, and a maximum accelerometer sensitivity of 6 g. The unit g is also known as g-force, and is the acceleration due to gravity and measured in units of m/s², where standing on the Earth’s surface represents 1g. The higher the sensitivity, the more accurate the reading is and the more easier it is to read the signal. The serial port terminal program GKTerm on the Ubuntu 10.04 was used to configure the IMU.

A python code on the Overo Fire reads in the data string from the IMU and uses pyserial to parse string into data. This data outputs values between 0 and 1024 from the 10-bit analog to digital converter. These values are then converted into voltage values by Equation 7.1 using a Gyro Sensitivity of 3.3 mV/deg/sec.
and Equation 7.2 using a accelerometer sensitivity of 800 mV/deg/sec at 1.5g. 

The GyroADCValue and ADCVal are both acquired from the sensor. In Equation 7.2, the VzeroG term is the voltage at zero g and must be taken out of the equation for proper readings.

\[
GyroRate = \frac{GyroADCValue \times 3.7 \times 10^{24}}{GyroSensitivity}
\]

(7.1)

\[
R_x = \frac{ADCVal \times 3.7 \times 10^{24} - V_{zeroG}}{AccelSensitivity}
\]

(7.2)

7.5 Compass

The Devantech Tilt Compensated Magnetic Compass purchased from Robot-shop shown in Figure 7.7 connects serially to the Overo Fire computer through ttyS0 at 9600 bps [31]. This compass gives the direction of the horizontal component of the predominant magnetic flux and only provides information of the magnetic field at its current location. It also reads the three axial components to measure roll and tilt. The compass is positioned horizontally on the center axis inside the pressure housing away from magnets, motors, and ferrous materials.

A pyserial python code reads in the serial data of four values and outputs the heading from a 16-bit value between 0 and 3599, pitch from a value between -85 to 85, and, and roll from a value between -85 to 85.
Figure 7.7: Compass

Figure 7.8 displays the board for the compass and the GPS. This board is powered by 3.3 volts and connects directly to the Tobi Expansion Board.

Figure 7.8: Sensor Board

7.6 Abort Signal to Emergency System

An abort message from the Overo Fire computer to the emergency system is for an emergency detected by the Overo Fire. Emergencies include the vehicle's
inability to correct the heading and stay on the navigation track.

The Overo Fire uses a general purpose input output (GPIO) pin to send out a signal. This pin is always set to 0 volts and when an emergency is detected, the Overo Fire is prompted to turn the GPIO pin high to 1.8 volts so it can be read by the Emergency System discussed in Chapter 9. The GPIO used for this signal is GPIO147, which is pin number 28 on the Tobi Expansion Board.

The Linux operating system allows for writing to the GPIO files to set the signal as an output or input, and a high or low. To initialize the GPIO147, a file for GPIO147 is created, an “out” is written to the file to initialize GPIO147 as an output, and then a “1” is written to the file in order to turn the pin high.

7.7 Flap Deflection Control

Wing flaps are controlled by a Hitec HS-311 Servo motor shown in Figure 7.9 purchased from Jameco Electronics [32]. This servo contains an electric motor, potentiometer, and electronics to translate the control signal to a position. The servo is powered by a 5 volt signal, and a logic level converter in Figure 7.10 is placed between the Overo Fire and the servo to increase signal logic from 1.8 volts to 5.0 volts.

A pulse width modulated signal (PWM) controls the servo. Figure 7.11 displays three examples of PWM signals. The servo motor expects a pulse every 20 ms, which is the period of the PWM signal. The amount of time the signal is high, also known as the length of the pulse, determines how far the motor turns. For example, a signal that is high for 1.5 ms allows the servo to turn to the neutral position of 90°. The servo moves to that position, holds, and resists moving out of that position while the pulse is repeated. If the signal
is less than 1.5 ms, the servo moves counterclockwise and if it is more than 1.5 ms, the servo moves clockwise.
Figure 7.11: Pulse Width Modulation (PWM)

The duty cycle in Equation 7.3 of a PWM signal is a ratio of the duration the signal is high ($\tau$) to the period ($T$) of the PWM signal.

$$DC = \frac{\tau}{T} \quad (7.3)$$

A typical servo motor operates at a frequency ($f$) of 50 Hz and is calculated using the following equation, using 20 ms as the signal period.

$$f = \frac{1}{T} \quad (7.4)$$

### 7.7.1 PWM Generation

The OMAP35x Applications Processor [26] made by Texas Instruments [33] on the Overo Fire has four general purpose timers to output PWM signals. PWM10 and PWM11 are used to control the wing flap deflection. Each timer has a base address and an offset address for each command.
Table 7.5: Timer Address Summary [26]

<table>
<thead>
<tr>
<th>Timer</th>
<th>Base Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIO147_PWM8</td>
<td>0x4903 E000</td>
</tr>
<tr>
<td>GPIO144_PWM9</td>
<td>0x4903 0000</td>
</tr>
<tr>
<td>GPIO145_PWM10</td>
<td>0x4808 6000</td>
</tr>
<tr>
<td>GPIO146_PWM11</td>
<td>0x4808 8000</td>
</tr>
</tbody>
</table>

Table 7.6 shows the register name and offset address needed to perform the functions to output a PWM signal. The physical address for TIMER10 and TIMER11 is the address offset shown in column three of Table 7.6 added to the address shown in Table 7.5.

Table 7.6: PWM Output Definitions [26]

<table>
<thead>
<tr>
<th>Command</th>
<th>Definition</th>
<th>Address Offset</th>
<th>Physical Address TIMER10</th>
<th>Physical Address TIMER11</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCLR</td>
<td>Stop/Start</td>
<td>0x024</td>
<td>0x48086024</td>
<td>0x48088024</td>
</tr>
<tr>
<td>TCRR</td>
<td>Counter Register</td>
<td>0x028</td>
<td>0x48086028</td>
<td>0x48088028</td>
</tr>
<tr>
<td>TLDR</td>
<td>Frequency</td>
<td>0x02C</td>
<td>0x4808602C</td>
<td>0x4808802C</td>
</tr>
<tr>
<td>TMAR</td>
<td>Duty Cycle</td>
<td>0x038</td>
<td>0x48086038</td>
<td>0x48088038</td>
</tr>
</tbody>
</table>

The following steps detail the steps to produce a PWM signal on the Overo Fire [34].

1. Disable the timer using the TCLR register.

2. All PWM pins on the Overo Fire are originally set as general purpose input/output (GPIO) pins. These pins must be changed to PWM by changing the PAD configuration register.

3. The frequency is controlled by the TLDR register. The clock frequency (FCLK) is used in the calculation of the value to write to TLDR. TIMER8
and TIMER9 have a frequency of 13 MHz and TIMER10 and TIMER11 have a frequency of 32 kHz which can be changed to 13 MHz. Equation 7.5 is used to calculate the value to write to the TLDR register.

\[ \text{TimerFrequency} = \frac{FCLK}{((0xFFFFFFFF - TDLR) + 1)} \]  

(7.5)

4. The duty cycle is controlled using the TMAR register. The number of settings in Equation 7.6 are the number of TMAR values there are available.

\[ \text{NumberOfSettings} = 0xFFFFFFFFE - TDLR \]  

(7.6)

5. From the Number of Settings, the TMAR register value for the PWM generated is calculated using Equation 7.7. The TLDR value for PWM generation is 0xFFFFFD80.

\[ \text{TMAR} = \text{TLDR} + (\text{NumberOfSettings} \times \text{DutyCyclePercentage}) \]  

(7.7)

The Number of Settings equation in Equation 7.6 is used to calculate servo motor resolution. According to Equation 7.6, the 32kHz clock allows for 638 duty cycle steps with 0 corresponding to 0% duty cycle and 683 corresponding to 100% duty cycle. Since this servo is only moving between -20° and 20°, the servo only needs a 6.4% to 8.4% duty cycle. A servo range of 6.4% to 8.4% duty cycle has approximately 12.76 step values to work with at a 32kHz frequency according to Equation 7.8. These calculations are shown in Appendix G.

\[ \text{Steps} = \left( \frac{\text{dutycyclerange}}{100} \right) \times \text{NumberOfSettings} \]  

(7.8)
This gives 12.76 values to work with, and the total of steps needed are 9 steps, including 4 up, 4 down, and 1 for neutral. According to Equation 7.9, the time in microseconds between servo steps, also known as the resolution, is 31.35 µs.

\[ \text{Resolution} = \frac{(High\text{PulseWidth} - Low\text{PulseWidth})}{\text{Steps}} \]  

(7.9)

### 7.7.2 Flap Deflection Test Results

After successful PWM generation out of the Overo Fire, different duty cycles were tested for servo movement. The following table displays the degree of upward and downward deflection, pulse width, and duty cycle needed to deflect the flap.

Table 7.7: Angle and TMAR value

<table>
<thead>
<tr>
<th>Angle</th>
<th>Pulse Width</th>
<th>Duty Cycle</th>
<th>TMAR Hex Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>20°</td>
<td>1.65 ms</td>
<td>8.4%</td>
<td>0xFFFFFDDB5</td>
</tr>
<tr>
<td>15°</td>
<td>1.59 ms</td>
<td>8.0%</td>
<td>0xFFFFFDDB3</td>
</tr>
<tr>
<td>10°</td>
<td>1.55 ms</td>
<td>7.9%</td>
<td>0xFFFFFDAA2</td>
</tr>
<tr>
<td>5°</td>
<td>1.49 ms</td>
<td>7.6%</td>
<td>0xFFFFFDAB0</td>
</tr>
<tr>
<td>0°</td>
<td>1.45 ms</td>
<td>7.5%</td>
<td>0xFFFFFDAF</td>
</tr>
<tr>
<td>-5°</td>
<td>1.40 ms</td>
<td>7.1%</td>
<td>0xFFFFFDAD</td>
</tr>
<tr>
<td>-10°</td>
<td>1.34 ms</td>
<td>6.8%</td>
<td>0xFFFFFDAB</td>
</tr>
<tr>
<td>-15°</td>
<td>1.31 ms</td>
<td>6.6%</td>
<td>0xFFFFFDAA</td>
</tr>
<tr>
<td>-20°</td>
<td>1.25 ms</td>
<td>6.4%</td>
<td>0xFFFFFDAB</td>
</tr>
</tbody>
</table>
Chapter 8

Vehicle Control

The Glider Overo Fire reads from all sensors to monitor the vehicle state and estimate the location of the vehicle. Figure 8.1 displays the sensors and servo motor board connected to the Overo Fire.

Figure 8.1: Glider Electronics
Overall, the control system of the glider includes the ability to:

- Monitor vehicle performance
- Log and store sensor data
- Adjust glide angle by pitch control using wing flaps
- Adjust heading and control roll using wing flaps
- Monitor the mission timer

The main idea of the glider control in this thesis involves using a wing flap concept to navigate and control the vehicle underwater. This uses the Python programming language on the Overo Fire that was discussed in Chapter 7. This control uses threading, which is a module in Python that allows two programs to run simultaneously within the same program. In particular, the wing test code that was created to test flap deflection, movement capability, and control before each dive, uses threading in Python to generate the PWM signals from the Overo Fire. In this case, threading is used to run a function to turn the left wing and an function to turn the right wing simultaneously.

The dive program used to implement a dive turns wing flaps down during the descent of the glider. The direction of turning the flaps varies depending on wing position and location of the center of gravity and center of buoyancy.

The compass on the glider measuring heading, pitch, and roll, takes an initial reading and compares it to the actual compass reading during the dive. The Overo Fire acquires data from sensors every 2 seconds, which is adequate due to the slow traveling velocity of gliders. Using the wing flaps, the vehicle corrects for a change in roll by moving one flap up and one flap down.
The time module in Python is used to record the start time, end time, and time elapsed and compares the actual dive time to the planned dive time. If the time exceeds the planned dive time and if the heading, pitch, and roll correction are not working properly, an abort message is sent to the emergency system. This abort message ignites the external interrupt on the microcontroller to release the drop weight and the vehicle ascends to the surface.
Chapter 9

Emergency System

The emergency system on the glider is a sensor board measuring temperature, humidity, and pressure inside the pressure housing. This system is a self-powered unit controlled by a microcontroller and powered by a battery pack. Figure 9.1 displays a diagram of the flow of emergency system and the completed circuit is displayed in Figure 9.2 If the vehicle undergoes a change in temperature, humidity, and pressure in the pressure housing beyond a specified threshold, and a time greater than the maximum dive time, the emergency system turns a servo motor 90° to release a drop weight. At this point the glider surfaces to gather GPS coordinates and send the coordinates through a Radio Frequency (RF) Modem.

9.1 Microcontroller PIC18LF45k80

The emergency system is programmed on a PIC18LF45k80, which is the low power version of the PIC18F45k80, operating at 3.3 volts. The PIC18LF45k80 supports a voltage range from 2.0 to 5.5 volts. The emergency system is powered
by a 5 volt power source to drive the pressure sensor and servo motor and a 3.3 volt source for the sensors and RF modem. The microcontroller is programmed in the C language using MPLAB and C18 compiler libraries, and the Microchip PicKit3 shown in Figure 9.3 for transferring the program to the microcontroller. Table 9.1 displays all features of the PIC18LF45k80. Each external pin has more than one internal function and registers determine the internal functions of each pin.

The microcontroller has an external oscillator of 4MHz to provide an accurate clock source to control the execution of instructions. Capacitors along the
crystal circuit allows for crystal oscillation shortly after power is applied to the microcontroller. The crystal frequency is divided by 4 to create the internal clock signal’s frequency ($F_{osc}$), also known as the number of instructions per second. The instructions on a microcontroller require one instruction cycle to execute. For this circuit, each instruction cycle is executed with a 1 MHz frequency, which is a period ($T_{osc}$) of 1 $\mu$s. Most instructions on a microcontroller require one instruction to execute, with one execution cycle equaling four clock cycles.
Table 9.1: Features of PIC18LF45k80 [35]

<table>
<thead>
<tr>
<th>Features</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Frequency</td>
<td>64 MHz</td>
</tr>
<tr>
<td>Program Memory (Bytes)</td>
<td>32K</td>
</tr>
<tr>
<td>Program Memory (Instructions)</td>
<td>16,384</td>
</tr>
<tr>
<td>Data Memory (Bytes)</td>
<td>3.6K</td>
</tr>
<tr>
<td>Interrupt Sources</td>
<td>32</td>
</tr>
<tr>
<td>I/O Ports Ports</td>
<td>A, B, C, D, E</td>
</tr>
<tr>
<td>Comparators</td>
<td>Two</td>
</tr>
<tr>
<td>Timers</td>
<td>Five</td>
</tr>
<tr>
<td>Capture/Compare/PWM Modules</td>
<td>Four</td>
</tr>
<tr>
<td>Serial Communications</td>
<td>1 MSSP and 2 Enhanced USARTs</td>
</tr>
<tr>
<td>12-Bit Analog-to-Digital Module</td>
<td>Eleven Input Channels</td>
</tr>
<tr>
<td>Instruction Set</td>
<td>75 Instructions</td>
</tr>
</tbody>
</table>

9.2 Sensors

Table 9.2 displays an overview of the sensors on the emergency system sensor board. Necessary data sheets for the sensors are located in Appendix I.

Table 9.2: Overview of Sensors

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Voltage</th>
<th>Current</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF</td>
<td>2.8-3.4 V</td>
<td>50 mA (Receive) 215 mA (Transmit)</td>
<td>Serial(TX2/RX2)</td>
</tr>
<tr>
<td>GPS</td>
<td>3.3-5 V</td>
<td>65 mA</td>
<td>Serial(TX1/RX1)</td>
</tr>
<tr>
<td>Temp/Hum Sensor</td>
<td>2.4-5.5 V</td>
<td>1 mA</td>
<td>Two-wire Digital</td>
</tr>
<tr>
<td>Pressure Sensor</td>
<td>5 V</td>
<td>2.5-5 mA</td>
<td>ADC</td>
</tr>
<tr>
<td>Servo</td>
<td>4.8-6.0 V</td>
<td>600 mA(stalled) 7.4 mA (stopped)</td>
<td>PWM</td>
</tr>
</tbody>
</table>

9.2.1 Temperature and Humidity Sensor

The SHT15 Temperature and Humidity Sensor is made by Sensiron [36] and the sensor with the breakout board was purchased from Sparkfun Electronics [27].
This fully calibrated sensor communicates through a digital two-wire interface. The SHT15 sensor has a serial clock input (SCK) and serial data (SDA) pin. The SCK pin allows for communication between the SHT15 sensor and the microcontroller. The DATA pin is a bidirectional pin that transfers data into and out of the sensor. The sensor requires two pull up resistors, one on the SCK pin and one on the SDA pin for proper operation. Sending a command to the sensor requires that DATA be valid on the rising edge of the SCK pin. During the transmission process of reading the data, bytes are received most significant byte (MSB) first. A 9th bit ends the byte transmission and the receiver acknowledges the byte transmission by driving the DATA line low and the SCK line high.

![Image of the SHT15 sensor](image_url)

Figure 9.4: Temperature and Humidity Sensor

### 9.2.2 Pressure Sensor

The pressure sensor is a temperature compensated and calibrated silicon pressure sensor MPX5999D from Freescale Semiconductor [37]. The sensor in Figure 9.5 inputs analog signals into the microcontroller which are converted digital signals by the analog-to-digital (A/D) converter. The PIC18LF45k80 has a 12-
bit A/D module with a voltage resolution of 0.73 mV measuring at 3.3 volts. This pressure sensor operates at 5 volts and has a maximum operating pressure of 1000 kPa, with 1 kPa equaling 0.145 psi. Because the microcontroller operates at 3.3 volts, a A/D reference voltage of 4.096 is configured for the A/D converter. This allows for a voltage resolution of 1 mV.

![Figure 9.5: Barometric Pressure Sensor](image)

The datasheet provides a transfer function for the voltage to pressure calculation. This sensor has a zero pressure offset of 0.20 volts and an error term to offset the pressure to 0 kPa when no pressure is applied. This pressure reading reads a negative value if a vacuum is applied and reads a value greater than 0 if a leak is detected in the pressure housing.

### 9.2.3 Runtime Timer

A timer is implemented on the PIC18LF45k80 to record the time of each dive of the glider. If the time exceeds a certain time, the glider enters the abort mode and releases the drop weight.
9.2.4 External Interrupt for Abort Message

An external interrupt is an external event that triggers a deviation from the normal program flow [38]. In this case, the Overo Fire sends a high signal to the microcontroller in an emergency. The microcontroller’s external interrupt responds to this event automatically. The external interrupt is configured for the rising edge of the signal on the RB0/INT0 pin on the microcontroller. When this pin turns high, the main program is interrupted to go into abort mode and release the drop weight.

9.3 Dropweight Mechanism

Figure 9.6 displays a computer-aided design (CAD) model and image of the drop weight mechanism in the glider. This contains the servo motor shown in Figure 7.9 that turns a 90° rotation to release a level arm and release the weight from the cylinder.

The length of each pulse determines how far a servo motor turns. The microcontroller turns a digital pin high and low to turn the servo motor 90°.

9.4 At Surface

The PIC18LK45k80 microcontroller has forms of serial communication such as the Universal Synchronous Asynchronous Receiver Transmitter (USART) and Universal Asynchronous Receiver transmitter (UART). During USART data transfer, serial data enters the microcontroller through the RX pin. The receiver block automatically shifts the data into the RSR register upon detection of a start bit. The RSR contents are transferred into the RCREG after recognition.
of a stop bit. The receive interrupt flag is set to 1, indicating that there is data in the RCREG. The RCIF bit is cleared to 0 after data is read from the RCREG. The data is extracted from the RCREG buffer in the same order as it is written. An interrupt is triggered by the arrival of asynchronous serial data. The UART is performed through software, using the C18 compiler libraries in MPLAB.

### 9.4.1 GPS

After the glider performs an emergency assent to the surface, it will gather a GPS fix at the surface at 4800 bps. The GPS sensor from Parallax Inc [28] previously shown in Figure 7.4 tracks up to 20 satellites, gathering a string of data following NMEA0183 V2.2 data protocol [29]. The microcontroller separates the data to obtain the date, time, latitude, and longitude and communicates to the GPS through the hardware USART, using the pins labeled RC6/RX1.
and RC7/TX1. The GPS data is set to the RF modem and is logged with the datalogger.

9.4.2 Xbee RF Modems

The XBee-PRO 900 MHz module in Figure 9.7 was purchased from Sparkfun Electronics [27] and developed by Digi International [39]. This modem uses the additional hardware USART pins available on the PIC18LF45k80 at 9600 bps. This modem has an outdoor distance of 1 mile and an indoor distance of 300 ft. The modem is used for point-to-point transmission from the vehicle to the support boat.

![RF Modem](image)

Figure 9.7: RF Modem
9.4.3 Datalogger

The OpenLog Datalogger in Figure 9.8 is from Sparkfun Electronics [27]. This device is low power and receives data serially at 9800 bps and saves data to an SD card. This device uses software UART on the microcontroller.

Figure 9.8: Datalogger
Chapter 10

Science Package

The science package for this glider is a self contained Star Oddi conductivity temperature and depth sensor displayed in Figure 11.6(a). Due to the small compact size, the sensor is placed in the housing in Figure 11.6(b) and positioned in the nose cone of the glider. The polyurethane plastic housing provides protection for the sensor under harsh environments and serves as a mounting mechanism to the glider. This sensor logs data throughout a dive and can be accessed after the dive using the SeaStar software and Star Oddi Communications Box shown in Figure 11.6(c).
Figure 10.1: Star Oddi Conductivity Temperature Depth (CTD) Sensor [40]
Chapter 11

Vehicle Testing, Results, and Optimization

The following chapter describes testing, test results, and vehicle optimization for this glider. The majority of the testing was completed in the Underwater Technology Lab and the Center of Corrosion and Biofouling Control at Florida Institute of Technology. The purpose of this testing is to verify that this system meets the performance and design discussed in this thesis. Each test is explained and detailed with test results and optimizations.

11.1 Pressure Test

The pressure housing was assembled and tested in the hyperbaric chamber shown in Figure 11.1 at the Center for Corrosion and Biofouling Control. This test was completed to verify the design calculations, test for leaks around the end caps, and test for leaks around the wet connectors.

Before the test, wet-mateable connectors were fitted into the end cap of the
pressure housing in Figure 11.2. An o-ring was placed in the groove and the pressure housing was tightened by bolts to create a seal. The pressure housing ready for testing is shown in Figure 11.3.

The pressure housing was pressurized to 60 m (197 ft) in the hyperbaric chamber for a total test time of 20 minutes with 5 minutes bottom time. After opening the chamber at the end of the test, small bubbles were seen around the end cap seals of the pressure housing, confirming water leakage. Due to the orientation of the pressure housing during the test, only a minimal amount of water leaked into the housing.
Figure 11.2: Wet Connectors

Figure 11.3: Sealed Pressure Housing

Optimizing the pressure housing for future use includes adding a secondary o-ring to provide backup to the primary o-ring and changing the o-ring size to create a better seal. The o-ring used in the design of the pressure housing did not fit to the machined groove exactly due to machine error. Sanding the faces of the end cap and the pressure housing will increase the smoothness, increase
the contact area, and decrease the amount of water leaking into the housing.

11.2 Emergency Drop Weight Test

The following aspects of the emergency system in Figure 11.4 were tested to make sure all sensors were reading properly, the servo motor controlling the drop weight responded correctly, and the RF receiver board in Figure 11.5 received the GPS coordinates.

The following steps were completed during testing:

- Turned a general purpose input output (GPIO) pin high on the Overo Fire and tested the external interrupt on the microcontroller. This external interrupt released the drop weight and the GPS coordinates were successfully sent through RF.
• A maximum dive time was set and the timer exceeded the maximum dive time. This caused the servo to turn and release the drop weight and the GPS coordinates were successfully sent through RF.

• The temperature, humidity, and pressure were increased on the sensors to exceed the threshold. This caused the servo to turn, release the drop weight and send GPS coordinates through RF.

All steps during the drop weight test were successful. It was noted that the GPIO coming out of the Overo Fire did not need to be a full 3.3 volts in order to ignite the external interrupt on the microcontroller. A signal of 1.8 volts and a ground signal are both wired from the Overo Fire to the microcontroller to alert the external interrupt.

### 11.3 Wing Control Test

The wing control test code evaluates wing control performance, the flexibility of the live hinge, and the mounting technique. In this test, a pulse-width mod-
ulated (PWM) signal from the Overo Fire turned the servos to create positive and negative deflection. The test is as follows:

- Move the Left and Right wing flap simultaneously to the servo angle following degrees: -5°, 5°, -10°, 10°, -15°, 15°, -20°, 20°. Each flap centers at 0° between each deflection angle.

- Move the flaps in opposite directions simultaneously. Move left wing 5°, 10°, 15°, 20° and the right wing -5°, -10°, -15°, -20°.

The results of the wing control test were equal for both servo motors. Early on in the test, the servo movement was accurate, but as small increments of error accumulated, the servo positions were slightly off. The servo seems to lose its position, get off track, and have to correct the position.

The servo motors move easier and more accurately at the larger angles of 15° to 20°. Since this was the case, the angles 15° and 20° are used in the final version of this wing control test. The higher deflection provides more camber, an increase in lift, and a glide angle similar to the current gliders. Figure 11.6 displays the upward and downward deflection of the wings.
Each flap was made out of a unidirectional piece of fiberglass to produce stiffness along the wing. However, the flap deflection angles showed different results at the root and the tip of the wing because the flap was deflected from the root area. Figure 11.7 and Figure 11.8 display the wing angle of deflection at the root of the wing and the angle of deflection at the tip of the wing for both the right and left wings.
Figure 11.7: Servo Angle and Flap Deflection of Right Wing

Figure 11.8: Servo Angle and Flap Deflection of Left Wing
Optimizations resulting from this wing test include operating the servos at a higher resolution, possibly using the higher clock frequency that can be used on the Overo Fire PMW pins. After building the wings, one wing came out softer and more flexible than the other. This caused the wing servo to stall at certain positions during the test. Using a worm gear motor system for flap deflection can provide a stronger and accurate position with greater longevity and the motor can be stopped while holding a flap deflection position.

11.4 Dive

The dive test began by turning the flaps down to imitate vehicle descent. At this point, the vehicle responded to changes in roll read by the compass. If the current roll value was greater than the initial roll value, the vehicle rolled to the left. If it was less than, the vehicle rolled to the right. The wings were moved opposite each other in order to correct for the rolling motion. At the maximum dive time, the vehicle flattens out the wings, and sends the high signal to the emergency system to drop the weight. This resembled the bottom of a dive and demonstrated proper communication with the microcontroller.

11.5 Structural Optimization

During the glider design process, it was determined that the glider fairing produces major vehicle drag. Through results from calculations of fairing drag, the teardrop size of the fairing should have a smaller diameter. Decreasing the diameter of the fairing from 0.35 m (13.62 in) to 0.17 m (6.82 in) decreases the fairing drag force from 4.23 N (0.95 lbf) to 1.82 N (0.41 lbf). Decreasing
the diameter increases the length of each wing, creating long slender wings to increase glider performance. Decreasing the diameter can have an adverse effect and decrease the amount of payload space in the glider. Using a fairing with a maximum diameter equal to the maximum diameter of the pressure housing will decrease drag and minimally decrease the payload space.
Chapter 12

Conclusions and Future Work

The principle aim of this work is to design and develop an autonomous underwater glider with focus on using external wing control surfaces for vehicle control. Underwater gliders today are used widely in oceanographic applications and have enormous ability to improve. This potential can be enhanced by the application of the methods presented in this thesis. The glider in this thesis provides the necessary foundation for glider research and future vehicle development. The following points detail some of the main ideas for future work on this vehicle.

- **Buoyancy Engine**  The glider buoyancy engine creates the majority of glider movement in the water and accounts for glider weight. Adding a buoyancy system to this glider allows for proper evaluation of the glider wings and wing flap performance.

- **Graphical User Interface (GUI)**  
  A graphic user interface allows for dive planning, setting the initial and final GPS coordinates, setting the speed, dive time, and emergency abort time. This software must have the ability to turn on and off the glider,
sensors, and motors from the computer user interface through an RF or wireless connection.

- **Enhancements to the Overo Fire**
  Enhancements to the Overo Fire includes drivers specific to this glider. The installation of drivers require extensive knowledge of open embedded Linux programming. For example, installing a driver or Python class for the pulse-width modulated (PWM) signal output allows the ease of use and a decrease in programming time.

- **Underwater Acoustic Tracking**
  A low-power acoustic transducer for glider tracking and glider location is necessary for the dives to locate the vehicle in case of a failure.

- **Pressure Sensor and Altimeter**
  A pressure sensor to measure depth and an altimeter to measure distance from the sea floor are important sensors to add to the glider. With the sensors, the control program can be modified to check for the depth of the glider. The glider can descend with flaps down while continuously checking the depth. When the glider becomes close to the maximum depth, it slowly levels out the wing flaps in order to minimize overshooting. Minimal overshooting is necessary since gliders are low-power and are not able to use the energy from correcting the overshoot.

- **Navigation Algorithm**
  Implementation of a navigation algorithm that uses Kalman Filtering and dead reckoning for underwater navigation is important. A Kalman Filter estimates the state of a dynamic system based on prior knowledge of the state and measurements from sensors on the vehicle. It uses values from
the inertial measurement unit (IMU) and time, to determine the speed of
the vehicle which is then converted into position.

- **Glider Model**
  Glider dynamics is a very important subject to underwater gliders. Creating a glider dynamics model specific to this glider using the hull, the wing airfoil, and the external wing control surfaces will evaluate and predict glider performance.

- **Water Tests**
  Any calculations performed on the vehicle and appendages must be verified through water tests. Once the whole vehicle is assembled, the center of gravity and center of buoyancy must be determined in order to find the location of each. Once this location is determined, the direction of flap deflection is determined based on the location of the flaps relative to the center of buoyancy and the center of gravity. The wing performance on the vehicle including lift, drag, and angle of deflection must be tested in the water test.

- **Modification of the Wing Control**
  A modification to the servo motor control mechanism includes a stepper motor and worm drive gear system. The stepper motor with worm drive gear system does not need to be continuously powered in order to stay in position. The worm gear acts as a brake and has the ability to turn the gear but the gear cannot turn the worm gear. This provides stable positioning of the flaps during a dive.

  Autonomous underwater gliders today have enormous ability to improve. The principle aim of this work is to design and develop an autonomous un-
nderwater glider with focus on using external wing control surfaces for vehicle control, which is one step toward the improvement of underwater gliders. With the additions of all the future work objectives, this glider has the ability to be the first fully inexpensive autonomous underwater glider at Florida Institute of Technology.
References


Appendix A

Python Equation Calculations and Library

The following Python code includes pressure hull calculations, wing lift and drag calculations, and creation of the glider dive profile.

%------------Calculation Output------------------------%

Density (slugs/ft^3): 1.9938
Kinematic Viscosity (ft^2/s): 1.1088e-05
Desired Velocity (m/s): 0.4
Maximum Design Depth (m): 1000
Maximum Design Pressure (psi): 445.835833333

PRESSURE HULL STRENGTH CALCULATIONS:
Maximum Pressure (psi): 11117.7741741
Maximum Depth (ft): 24936.923748
Rated Depth (ft): 12468.4618674
Rated Depth (m): 3801.0
Volume Lost to compression (in^3): 2.73893090215
Stress on End Plate (psi): 5575.2699783
Minimum Thickness (in): 0.717160132396
FAIRING LIFT AND DRAG CALCULATIONS:
Fairing Reynolds Number: 764189.51419
Fairing Coefficient of Drag: 0.549398087268
Fairing Drag Force (lbf): 0.954567887733
Fairing Aspect Ratio: 5.6880733945

WING AND TAIL CALCULATIONS:
Wing Coefficient of Drag: 0.0403697926958
Wing Coefficient of Lift: 0.708463899471
Wing Reynolds Number: 66558.4415584
Wing Aspect Ratio: 5.31555555556
Wing Coefficient of Drag (Raymer): 0.0197802472209
Wing Coefficient of Lift (Raymer): 0.347534374043
1/d Ratios: 78.2258064516
L/D Ratios: 16.4492061635
Lr/Dr Ratios: 17.5697987848
Total Lift per wing (lbf): 0.501511333451
Total Drag per wing (lbf): 0.0285439916479
Reynolds Number for Tail: 39402.5974026
Coefficient of Drag for Tail: 0.0149
Drag force on the Tail (lbf): 0.00355046980034

VEHICLE PERFORMANCE:
Total force of drag for the vehicle (lbf): 1.01520634083
Glide Angle (Degrees): 45.345879785
Required Change of Vol (in^3): 110.863839382
Time to Dive (min): 3514.38265251
Time to Surface (min): 7028.76530503
Time to Surface (hr): 1.95243480695
Cycle Range (m): 1975.99764021
Vertical Speed (m/s): 0.284472153106
Horizontal Speed (m/s): 0.28105815621

##Calculations for glider performance of an autonomous
#Underwater Glider. Uses the file GliderDesign.py for equations.
#Completed by Cheryl Skibski, Ocean Engineering,
#Florida Institute of Technology

import sys
from math import *
from pylab import *
sys.path.append('./')
import GliderDesign
### Design Parameters

- \( \rho = 1.9938 \text{ slugs/ft}^3 \)
- \( k_{\text{visc}} = 0.000011088 \text{ ft}^2/\text{s} \)
- \( v_{\text{velocityms}} = 0.40 \text{ m/s} \)
- \( v_{\text{velocity}} = v_{\text{velocityms}} \times 3.28 \text{ ft/s} \)
- \( \text{MaxDepth} = 1000.0 \text{ m} \)
- \( \text{MaxDepthft} = \text{MaxDepth} \times 3.28 \text{ ft} \)
- \( \text{MaxPress} = \rho \times 32.2 \times \text{MaxDepth} / 144 \text{ psi} \)
- \( \text{MaxDepthIn} = \text{MaxDepth} / 12 \text{ in} \)
- \( F_S = 2.0 \)
- \( g = 32.2 \text{ ft/s}^2 \)

```
print 'DESIGN PARAMETERS:
print 'Density (slugs/ft^3): ' + str(rho)
print 'Kinematic Viscosity (ft^2/s): ' + str(kvisc)
print 'Desired Velocity (m/s):' + str(velocityms)
print 'Maximum Design Depth (m): ' + str(MaxDepth)
print 'Maximum Design Pressure (psi): ' + str(MaxPress) + '\n'
``` 

### Pressure Hull

```
# PRESSURE HULL STRENGTH CALCULATIONS:
PHullMod = 10000000.0  # psi
PHullPoissonRatio = 0.33  # Dimless
PHullSigmaY = 40000.0  # psi
PHullID = 6.125  # in
PHullL = 20.0  # in
PHullEndThick = 0.75  # in
PHullCylt = 0.25  # in
PHullEndRadius = (PHullID / 2)  # in
PHullOD = PHullID + 2 * PHullCylt  # in
PHullMeanD = (PHullID + PHullOD) / 2.0  # in
PHullNumRings = 4  # in
PHullRingH = 1.0  # in
PHullRingW = 0.5
PHullRingSpace = PHullL / (PHullNumRings - 1)  # in
PHullXi = 0.5 * PHullRingH  # in
PHullRingA = PHullRingW * PHullRingH  # in
PHullRingI = PHullRingW * PHullRingH**3 / 12  # in^4
PHullRingCylI = GliderDesign.Il(PHullRingA, PHullCylt, PHullRingSpace, PHullRingI, PHullXi)  # in^4
PHullMaxPressure = GliderDesign.Pcr(PHullMeanD, PHullCylt,
```
```python
PHullRingW, PHullRingA, PHullPoissonRatio, PHullRingSpace, PHullL, PHullSigmaY, PHullMod, PHullRingCylI
MaxDepthPhull = PHullMaxPressure * 144 / (rho * 32.2)
PHullCompress = GliderDesign.CompressCyl(PHullMeanD, PHullCylt, PHullRingSpace, PHullRingA, PHullRingW, PHullPoissonRatio, PHullMod)
PHullVolLost = GliderDesign.DeltaVCyl(PHullCompress, PHullNumRings - 1, 3801)
EndPlateStress = GliderDesign.PlateStressFix(MaxPress, PHullEndRadius, PHullEndThick, PHullPoissonRatio)
MaxThick = GliderDesign.PlateTFS(1000 * 3.28 * rho * 32.2 / 144, 6.125 / 2, PHullPoissonRatio, PHullSigmaY, FS)

print 'Maximum Pressure (psi): ' + str(PHullMaxPressure)
print 'Maximum Depth (ft): ' + str(MaxDepthPhull)
print 'Rated Depth (ft): ' + str(MaxDepthPhull / FS)
print 'Rated Depth (m): ' + str(floor(MaxDepthPhull / FS / 3.28))
print 'Volume Lost to compression (in^3): ' + str(PHullVolLost)
print 'Stress on End Plate (psi): ' + str(EndPlateStress)
print 'Minimum Thickness (in): ' + str(MaxThick) + '

###------------------Fairing Lift and Drag----------------------###
print 'FAIRING LIFT AND DRAG CALCULATIONS:'
FairingL = 77.5 / 12.0 # ft
FairingD = 13.625 / 12 # ft
FairingA = (pi/4) * FairingD**2 # ft^2
LDr = FairingL / FairingD
FairingRe = GliderDesign.Re(velocity, FairingL, kvisc)
FairingCD = GliderDesign.CDb(LDr, GliderDesign.Cfb(FairingRe))
FairingDrag = GliderDesign.Drag(FairingA, FairingCD, velocity, rho)
FairingAR = FairingL / FairingD

print 'Fairing Reynolds Number: ' + str(FairingRe)
print 'Fairing Coefficient of Drag: ' + str(FairingCD)
print 'Fairing Drag Force (lbf): ' + str(FairingDrag)
print 'Fairing Aspect Ratio: ' + str(FairingAR) + '

### --------------- Wing Calculations ------------------------###
print 'WING CALCULATIONS:'
# Wing includes the flap attached
WingSpan = 2.99 # ft
WingSweep = 35.0 # degrees
WingTipChord = 0.375 # ft
WingRootChord = 0.75 # ft
```

112
WingTaper = WingTipChord / WingRootChord  # Dimless
# NACA 63-010 Modified for maximum thickness at 50%
WingCl = 0.97  # Dimless
WingCd = 0.0124  # Dimless
Wingalpha = 0.0  # Degrees - Angle of attack
Wingalpha0 = -9.2  # Degrees - Angle of attack zero lift

WingArea = (WingRootChord + WingRootChord * WingTaper) / 2 * WingSpan
WingAspect = GliderDesign.ARw(WingSpan, WingArea)
WingOswaldEff = GliderDesign.OswaldEff(WingArea, WingSweep)
Winga0 = GliderDesign.a0(WingCl, Wingalpha, Wingalpha0)
Winga = GliderDesign.a(Winga0, WingOswaldEff, WingAspect)
WingRe = GliderDesign.Re(velocity, ((WingRootChord + WingRootChord * WingTaper) / 2), kvisc)
WingCL = GliderDesign.CLw(Winga, Wingalpha, Wingalpha0)
WingCD = GliderDesign.CDw(WingCd, WingCL, WingOswaldEff, WingAspect)

print 'Wing Coefficient of Drag: ' + str(WingCD)
print 'Wing Coefficient of Lift: ' + str(WingCL)
print 'Wing Reynolds Number: ' + str(WingRe)
print 'Wing Aspect Ratio: ' + str(WingAspect)

###------------------Raymer Wing Calculations-------------###
# The Raymer method takes into account the fairing for
# the wing calculations
Rada0 = GliderDesign.a0(WingCl, Wingalpha*pi/180, Wingalpha0*pi/180)
Wingaray = GliderDesign.araymer(Rada0, WingAspect, WingSpan, WingRootChord, WingSweep, FairingD, WingTaper)
WingCLr = GliderDesign.CLw(Wingaray, Wingalpha*pi/180, Wingalpha0*pi/180)
WingCDr = GliderDesign.CDwrRaymer(WingCd, WingCLr, WingOswaldEff, WingAspect, WingRe, WingArea, WingSweep*pi/180, WingRootChord, WingTaper, FairingD, WingSpan)
TotalLift = GliderDesign.Lift(WingArea, WingCLr, velocity, rho)
TotalDrag = GliderDesign.Drag(WingArea, WingCDr, velocity, rho)

print 'Wing Coefficient of Drag (Raymer): ' + str(WingCDr)
print 'Wing Coefficient of Lift (Raymer): ' + str(WingCLr)
print 'L/d Ratios: ' + str(WingCl/WingCd)
print 'L/D Ratios: ' + str(WingCL/WingCD)
print 'Lr/Dr Ratios: ' + str(WingCLr/WingCDr)
print 'Total Lift per wing (lbf):' + str(TotalLift/2)
print 'Total Drag per wing (lbf):' + str(TotalDrag/2) + '
'
###------------------Tail Calculations ----------------##
print 'TAIL CALCULATIONS:
#Naca 63-010 Modified where maximum thickness is at 50%
TailRootChord=0.458
TailTipChord=0.208
TailTaper=TailTipChord/TailRootChord
TailHeight=0.417
TailA=(TailRootChord+TailRootChord*TailTaper)/2*TailHeight
TailCd=0.0149 #drag for Tail infinite
TailRe=GliderDesign.Re(velocity, ((TailRootChord+TailRootChord*TailTaper)/2), kvisc)
DragF=GliderDesign.Drag(TailA, TailCd, velocity, rho)

print 'Reynolds Number for Tail: ' + str(TailRe)
print 'Coefficent of Drag for Tail:' + str(TailCd)
print 'Drag force on the Tail (lbf): ' + str(DragF) + '
'
###---Total Lift and Drag for Entire Vehicle----------------##
print 'VEHICLE PERFORMANCE:
LiftAC=WingArea*WingCLr
DragAC=(WingArea*WingCDr)+(FairingA*FairingCD)+(TailA*TailCd)
GlideAngle=GliderDesign.GlideAngleGenS(LiftAC, DragAC)
GlideAngleDeg=degrees(GlideAngle)
TotalD=TotalDrag+DragF+FairingDrag

print 'Total force of drag for the vehicle (lbf):' + str(TotalD)
print 'Glide Angle (Degrees): ' + str(GlideAngleDeg)

### -------------- Vehicle Performance- -------------------##
HorizSpeed=velocity*cos(radians(GlideAngleDeg)) #ft/s
VertSpeed=velocity*sin(radians(GlideAngleDeg))
DeltaW=GliderDesign.WforV(rho, velocity, WingArea, WingCL, WingCD, radians(GlideAngle), FairingA,FairingCD, TailA, TailCd)*2
TimeToDive=MaxDepthft/VertSpeed #s
TimeToSurface=2*TimeToDive #s
Range=TimeToSurface*HorizSpeed #ft

print 'Required Change of Vol (in^3):' + str(((DeltaW/(gravity*rho))*12**3)
print 'Time to Dive (min): ' + str(TimeToDive)
print 'Time to Surface (min): ' + str(TimeToSurface)
print 'Time to Surface (hr): ' + str(TimeToSurface/3600)
print 'Cycle Range (m): ' + str(Range/3.28)
print 'Vertical Speed (m/s): ' + str(VertSpeed*0.3048)
print 'Horizontal Speed (m/s): ' + str(HorizSpeed*0.3048) + '\n'

#---------------------Dive Profile-------------------#

Profile=GliderDesign.Dive(LiftAC, DragAC, DeltaW, MaxDepth)
figure(1)
plot(Profile[1],Profile[2],Profile[1],Profile[3])
plot([0.0, 2000.0],[0.0,0.0], 'g') # plot surface of ocean green
legend(('Dive Path', 'Water Surface'), loc=4)
suptitle('Glider Dive Path') # add a title to the figure
xlabel('Range (m)')
ylabel('Depth (m)')
ylim(ymax=100)
ylim(ymin=-1100)
show() # Display Graph

###---Plot Glide angle (degrees) vs. wing flap deflection----###
# Get values of Cl and Cd using flaps deflections of 0, 5, 10, #15 degrees, plot
# Using a wing reynolds number = 66600
# Values taken from Javafoil
WingCl_0=-0.017    # Dimless
WingCd_0=0.01385   # Dimless
Wingalpha_0_0=0.0  # Degrees - Angle of attack
Wingalpha0_0=0.1465

WingCl_5=0.326     # Dimless
WingCd_5=0.01512   # Dimless
Wingalpha_5=0.0    # Degrees - Angle of attack
Wingalpha0_5=-2.94

WingCl_10=0.654    # Dimless
WingCd_10=0.02025  # Dimless
Wingalpha_10=0.0   # Degrees - Angle of attack
Wingalpha0_10=-6.03

WingCl_15=0.964    # Dimless
WingCd_15=.01167   # Dimless
Wingalpha_15=0.0  #Degrees - Angle of attack
Wingalpha0_15=-9.13

WingCl_20=1.237  #Dimless
WingCd_20=0.01686  #Dimless
Wingalpha_20=0.0  #Degrees - Angle of attack
Wingalpha0_20=-12.27

deflection=[0, 5, 10, 15, 20]
print deflection
WingCl=[WingCl_0, WingCl_5, WingCl_10, WingCl_15, WingCl_20]
WingCd=[WingCd_0, WingCd_5, WingCd_10, WingCd_15, WingCd_20]
Wingalpha=[Wingalpha_0, Wingalpha_5, Wingalpha_10, Wingalpha_15, Wingalpha_20]
Wingalpha0=[Wingalpha0_0, Wingalpha0_5, Wingalpha0_10, Wingalpha0_15, Wingalpha0_20]
WingCLr=[0.0, 0.0, 0.0, 0.0, 0.0]
i=0
GlideAngleFinal=[0.0, 0.0, 0.0, 0.0, 0.0]

for i in xrange(0,5):
    i=i+1
    Rada0=GliderDesign.a0(WingCl[i-1],Wingalpha[i-1]*pi/180, Wingalpha0[i-1]*pi/180)
    Wingaray=GliderDesign.araymer(Rada0,WingAspect,WingSpan, WingRootChord, WingSweep, FairingD, WingTaper)
    WingCLr=GliderDesign.CLw(Wingaray, Wingalpha[i-1]*pi/180, Wingalpha0[i-1]*pi/180)
    WingCDr=GliderDesign.CDwRaymer(WingCd[i-1],WingCLr, WingOswaldEff,WingAspect,WingRe,WingArea,WingSweep*pi/180, WingRootChord,WingTaper,FairingD,WingSpan)
    LiftAC=WingArea*abs(WingCLr)
    DragAC=(WingArea*WingCDr)+(FairingA*FairingCD)+(TailA*TailCd)
    GlideAngle=GliderDesign.GlideAngleGenS(LiftAC, DragAC)
    GlideAngleDeg=degrees(GlideAngle)
    GlideAngleFinal[i-1]=int(GlideAngleDeg)
print GlideAngleFinal

figure(2)
plot(GlideAngleFinal, deflection)
suptitle('Flap Deflection vs. Glide Angle')
ylabel('Deflection Angle (degrees)')
xlabel('Glide Angle (degrees)')
from math import *
from types import *

###----------------------- Wing Calculations --------##

def Lift(S, CL, v, rho):
    'Returns Lift'
    return 0.5*rho*S*CL*v**2

def Drag(S, CD, v, rho):
    'Returns Drag'
    return 0.5*rho*S*CD*v**2

def WforV(rho, v, Aw, CLw, CDw, theta, Ab, CDb, At, CDt):
    'Returns the Net Weight Required for a glide'
    #This function is based on a force balance in glide,
    #assuming the axis of
    #the body is exactly parallel to the glide path.
    #rho=Density of fluid. v=Velocity along glide path.a
    #Aw=Planform Area of
    #wings. CLw=Coefficient of Lift of Wings.
    #CDw=Coefficient of Drag of
    #wings. theta=Glide angle in radians.
    #Ab=Cross-sectional Area of Body.
    #CDb = Coefficient of Drag of the Body
    return 0.5*rho*v**2* (Aw*CLw*cos(theta)+
    Aw*CDw*sin(theta)+Ab*CDb*sin(theta)+
    At*CDt*sin(theta))

def WforVGenS(rho, v, LAC, DAC, theta):
    'Returns the Net Weight Required for a glide'
    return 0.5*rho*v**2* (LAC*cos(theta)+DAC*sin(theta))

def VforW(rho, W, Aw, CLw, CDw, theta, Ab, CDb):
    'Returns the Net Weight Required for a stable glide'
    #This function is based on a force balance in glide,
#assuming the axis of
the body is exactly parallel to the glide path.
#rho=Density of fluid. v=Velocity along glide path.a
#Aw=Planform Area of
#wings. CLw=Coefficient of Lift of Wings.
#CDw=Coefficient of Drag of
#wings. theta=Glide angle in radians.
#Ab=Cross-sectional Area of Body.
#CDb= Coefficient of Drag of the Body
return sqrt(2*W/(rho*(Aw*CLw*cos(theta)+
#Aw*CDw*sin(theta)+Ab*CDb*sin(theta)))))

def VforWGenS(rho,W,LAC,DAC,theta):
    #Returns the Net Weight Required for a stable glide at
    #a Specified Velocity'
    return sqrt(2*W/(rho*(LAC*cos(theta)+DAC*sin(theta))))

def GlideAngle(Aw,CDw,CLw,Ab,CDb):
    'Returns Angle of Glide in Radians'
    #Aw = Planform area of wings.
    #CDw=Coefficient of Drag of Wings.
    #CLw=Coefficient of Lift of Wings.
    #Ab=Cross section Area of Body.
    #CDb=Coefficient of Drag of the Body
    return atan((Aw*CDw+Ab*CDb)/(Aw*CLw))

def GlideAngleGenS(LAC,DAC):
    'Returns Angle of Glide in Radians'
    return atan((DAC)/(LAC))

def ARw(b,Aw):
    'Returns the Aspect Ratio of Wings'
    #b=wingspan. Aw=Planform Area of Wings
    return b**2/Aw

def a0(Cl,alpha,alpha0):
    'Returns the Lift Slope Coefficient for
    an Infinite Wing'
    #Cl=Coefficient of Lift of infinite wing
    #(usually from profile data).
    #alpha=angle of attack where Cl is calculated.
    #alpha0=angle of attack for
    #zero lift
return Cl/(alpha-alpha0)

def a(a0,e,ARw):
    'Returns the Lift Slope Coefficient for a
    Finite Wing in Cl/Degree'
    #a0=infinite wing lift slope coefficient.
    #e=oswald efficiency factor.
    #ARw=aspect ratio of wings
    return a0/(1+(57.3*a0)/(pi*e*ARw))

def araymer(a0,ARw,Span, Crr, Sweep, d, Taper):
    'Returns the Lift Slope Coefficient for a
    Finite Wing in Cl/Radian
    according to Raymer'
    # BE SURE!! That a0 is in radians!!!
    #Otherwise this WILL FAIL!!!
    # It'll actually return a value, but that
    #value will be COMPLETELY wrong
    Crx=Crr*(1-(1-Taper)/Span)*d
    #Crx=Crr-(Crr-Crr*Taper)/(.5*Span)*0.5*d
    Sx=0.5*(Crx+Crr*Taper)*(Span-d)
    Sr=0.5*(Crr+Crr*Taper)*Span
    Beta=1.0
    eta=a0/(2*pi)
    F=1.07*(1+d/Span)**2
    if F>0.98:
        F=0.98
    return 2*pi*ARw/(2+sqrt(4 + (ARw**2*Beta**2)/eta**2 * 
                       (1+ tan(Sweep*pi/180)**2/(Beta**2)))) * (Sx/Sr)*F

def CDwRaymer(Cdw,CLw,e,ARw,Re,Sr,Sweep,Crr,Taper,d,Span):
    #return Cdw+(CLw**2)/(pi*ARw*e)+Cfw(Re)
    FF=(1+ 0.6/0.5*0.1 +100*0.1**4)*(1.34*1**
       0.18*cos(Sweep)**.28)
    Crx=Crr*(1-(1-Taper)/Span)*d
    #Crx=Crr-(Crr-Crr*Taper)/(.5*Span)*0.5*d
    Sx=0.5*(Crx+Crr*Taper)*(Span-d)
    Sr=0.5*(Crr+Crr*Taper)*Span
    Sw=2.0*(1+0.2*0.1)*Sx
    Q=1.0
    Cf=1.328/sqrt(Re)
    return (CLw**2)/(pi*ARw*e)+Cdw#Cf*FF*Q*Sw/Sr
def CLw(a, alpha, alpha0):
    'Returns the Coefficient of Lift for a Finite Wing'
    # a=finite wing lift slope coefficient.
    # alpha=angle of attack in degrees
    return a*(alpha-alpha0)

def CDw(Cdw, CLw, Oswald, ARw):
    'Returns the Total Coefficient of Drag of a Wing'
    # Cdw=Coefficient of Drag of infinite wing.
    # CLw=Coefficient of Lift of
    # finite wing. Oswald=Oswald efficiency factor.
    # ARw=Aspect Ratio of Wings
    # Cfw=Frictional Coefficient of Wing
    # return Cdw+(CLw**2/(pi*Oswald*ARw))+Cfw
    return (CLw**2/(pi*Oswald*ARw))+Cdw

def OswaldEff(S, LESweep):
    'Returns an Empirically Estimated Value for
    Oswald Efficiency Factor'
    # S=Wing Planform Area.
    # LESweep=Leading Edge Sweep in Degrees.
    if LESweep<=30:
        Oswald=1.78*(1-0.045*S**0.68)-0.64
    else:
        Oswald=4.61*(1-0.045*S**0.68)*
        cos(LESweep*pi/180)**0.15-3.1
    if Oswald>0.98:
        Oswald=0.98
    return Oswald

#------------------ Body Calculations------------------#
def Re(v, L, nu):
    'Returns Reynolds Number'
    # v=velocity. L=Characteristic Length.
    # nu=kinematic viscosity
    return v*L/nu

def Drag(S, CD, v, rho):
    'Returns Drag'
    return 0.5*rho*S*CD*v**2

def Cfb(Re):
    'Returns ITTC57 Friction Coefficient of a Body'
#Re=Reynolds Number  
return 0.075/(log10(Re)-2)

def CDb(LDr,Cfb):
    'Returns Total Coefficient of Drag of a Streamlined Body'
    #L=Length of Body.  D=Diameter of Body.
    #Cfb=Frictional Coefficient of
    #Body.
    return Cfb*(3*(LDr)+4.5*sqrt(LDr)+21*((1/LDr)**2))

###------------------Structural Calculations---------------###

def InstabPcr(E,h,Dm,Il,Ll,L):
    'Returns the Critical Pressure for General
    Instability of the Pressure Hull'
    # E=Modulus of elasticity of hull material.
    #h=Thickness of hull shell
    # Dm=Mean Diameter of Hull Shell.
    #Il=Moment of Inertia of hull’s frame/shell
    # combination.  Ll=Length between stiffeners.
    #L=Length between bulkheads
    # or similar structures--
    #likely the overall length between endcaps.
    R=Dm/2  
m=(pi*R/(L))  
t=h
    Pcr=[]
    for n in xrange(1,20):
        PcrOfn=(E*t/R) * ( m**4/((n**2+m**2/2-1)*(n**2+m**2)) ) +
                ((n**2-1)*E*Il)/(R**3*Ll)
        Pcr.append(PcrOfn)
    return min(Pcr)

def Il(A,h,Ll,I,Xi):
    'Returns the Moment of Inertia of the Hull
    Frame-Shell Combination'
    return (A*(Xi+(h/2.0)**2))/(1+A/(Ll*h))+I+(Ll*h**3)/12.0
    #A*(F+(t/2))^2)/(1+F/t)+I+(F^2t^3)/12

def BucklePcr(E,h,Dm,v,Ll):
    'Don’t use this function! It’s not working!!'
    'Returns Failure Pressure Due to Inter-Frame Buckling'
    #E=Modulus of Elasticity of Hull.  h=Hull Thickness.
    #Dm=Mean Diameter of
# Hull Shell. v = Poisson’s Ratio of Hull.
# Ll = Length Between Frames.
# return (2.42*E/((1-v**2)**(3.0/4.0)))*
# ((h/Dm)**(5.0/2.0)/(L/Dm)-0.45*sqrt(h/Dm))

return ((2.42*E)/((1-v**2)**(3/4)))*
((h/Dm)**(5/2))/((Ll/Dm-0.45*(h/Dm)**0.5))

def YieldPcr(Dm,h,b,Af,v,Ll,sigmay):
    'Returns the Yield Pressure for a ring-stiffened cylinder'
    # Dm = mean hull diameter. h = hull thickness.
    # b = reinforcing frame width.
    # Af = cross-sectional area of reinforcing frame.
    # v = Poisson’s ratio of hull material. Ll = length between centers of reinforcing frames.
    # sigmay = hull material yield strength.
    t = h
    R = Dm/2.0
    B = b*t/(Af+b*t)
    theta = 10.0*(12.0*(1.0-v**2.0))**(0.25)*(t**2.0/(Af+b*t))
    N = (cosh(theta)-cos(theta))/(sinh(theta)+sin(theta))
    Beta = (11.0*N/sqrt(50.0*t/R))*(t**2/(Af+b*t))
    H = 1.0*3*sinh(theta/2.0)*cos(theta/2.0) +
      cosh(theta/2.0)*sin(theta/2.0)
    / (sinh(theta)+sin(theta))
    return sigmay*(t/R)/(1+H*((0.85-B)/(1+B)))

def Pcr(Dm,h,b,Af,v,Ll,L,sigmay,E,Il):
    Pressure = []
    Pressure.append(YieldPcr(Dm,h,b,Af,v,Ll,sigmay))
    Pressure.append(BucklePcr(E,h,Dm,v,Ll))
    Pressure.append(InstabPcr(E,h,Dm,Il,Ll,L))
    return min(Pressure)

def BucklePcrUnStiff(E,v,h,Dm):
    'Returns the Critical Buckling Pressure for an unstiffened cylinder'
    # E = Young’s Modulus of Hull. v = Poisson’s Ratio of Hull.
    # Dm = Mean Diameter
    # of Hull. h = hull shell thickness
    return 2*E/(1-v**2)*(h/Dm)**3

def YieldPcrUnstiff(sigmay,h,Dm):
    122
'Returns the Yield Pressure of an Unstiffened Cylinder'
# sigmay = Hull Material Yield Strength.
# h = Hull shell thickness. Dm = Mean diameter
return 4/sqrt(3.0)*sigmay*h/Dm

def PlateStressSimp(p,r,t,v):
    'Estimates Stress in a circular flat plate with Simply Supported Edges'
    Mc = p*r**2*(1.0/4)*(1-(1-v)/4)
    return (6*Mc)/(t**2)

def PlateStressFix(p,r,t,v):
    'Calculates Stress in a circular flat plate with Fixed Edges'
    Mc = p*(r**2)*(1+v)*(1/16.0)
    Mra = -p*(r**2)/8
    sigma = [6*Mc/(t**2), 6*Mra/(t**2)]
    return max(abs(sigma[0]), abs(sigma[1]))

def PlateT(p,r,v,sigma):
    'Calculates the minimum thickness of a circular flat plate to resist given stress'
    return max(sqrt(3*p*r**2*(1+v)/(8*sigma)), sqrt(3*p*r**2/(4*sigma)))

def PlateTFS(p,r,v,sigma,FS):
    'Calculates the minimum thickness of a circular flat plate to resist given stress by FS'
    return max(sqrt(3*p*r**2*(1+v)/(8*sigma/FS)), sqrt(3*p*r**2/(4*sigma/FS)))

def CompressCyl(Dm,h,Lf,Af,b,v,E):
    'Returns the Cubic Inch Change in Volume of a Hull Section per Foot of Depth'
    # Dm = Hull Mean Diameter. h = Shell Thickness.
    # Lf = Center to Center Frame
    # Spacing. Af = X-sectional Area of Reinforcing Frame.
    # b = Width of Frame.
    R = Dm/2
    t = h
    L = Lf - b
    theta = 3*(1-v**2)**(1.0/4.0)*L / sqrt(R*t)
    N = (cosh(theta) - cos(theta))/(sinh(theta) + sin(theta))
    return 64.0/144.0 * (pi*R**3*Lf*(2-v)**2)/(2*E*t) *
\[
\left( 1+(1-v^2)/(2-v^2) - 1/(L_f/A_f + \theta/(2N)) \right)
\]

def DeltaVCyl(ComCyl, Nsec, Depth):
    'Returns volume lost in cubic inches due to cylinder compression'
    # ComCyl = Cylinder Compression Constant.
    # Nsec = Number of inter-stiffener sections.
    # Depth in feet.
    return ComCyl * Nsec * Depth

def Dive(Lift, Drag, deltaW, maxdepth):
    DProfile = [[0.0], [0.0], [0.0], [0.0]]  # [time, X, depth, eta]
    i = 0
    depth = 0.0
    eta = 0.0
    step = 1.0
    rho = 1025
    glide = True
    while (depth < eta or glide == True):
        i = i + 1
        Angle = GlideAngleGenS(Lift, Drag)
        Velocity = VforWGenS(rho * .00194, (deltaW * 1025.0 / rho) / 2.0, Lift, Drag, Angle)
        Velocity = Velocity / 3.28
        Vh = Velocity * cos(Angle)
        Vv = Velocity * sin(Angle)
        time = DProfile[0][i-1] + step
        if glide == True:
            depth = DProfile[2][i-1] - Vv * step
        else:
            depth = DProfile[2][i-1] + Vv * step
        if depth > eta:
            depth = eta
        X = DProfile[1][i-1] + Vh * step
        DProfile[0].append(time)
        DProfile[1].append(X)
        DProfile[2].append(depth)
        DProfile[3].append(eta)
        if depth < -1 * maxdepth:
            glide = False
    return DProfile
Appendix B

CAD Drawings

The following are CAD drawings from Autodesk inventor showing pieces machined for this thesis.
Cylinder

Pressure Housing and O-Ring

<table>
<thead>
<tr>
<th>DRAWN</th>
<th>6/11/2011</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHECKED</td>
<td>TITIE</td>
</tr>
<tr>
<td>QA</td>
<td>Pressure Housing and O-Ring</td>
</tr>
<tr>
<td>MFG</td>
<td></td>
</tr>
<tr>
<td>APPROVED</td>
<td></td>
</tr>
<tr>
<td>SIZE</td>
<td>DWG NO</td>
</tr>
<tr>
<td>REV</td>
<td></td>
</tr>
<tr>
<td>SCALE</td>
<td>SHEET 1 OF 1</td>
</tr>
</tbody>
</table>
Glider Wing

Thickness = 0.70

DRAWN 6/11/2011
CHECKED
QA
MFG
APPROVED

SIZE DWG NO REV

SCALE

Sheet 1 of 1
Maximum Thickness = 0.50

<table>
<thead>
<tr>
<th>DRAWN</th>
<th>6/11/2011</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE</td>
<td>Glider Tail</td>
</tr>
<tr>
<td>SIZE</td>
<td></td>
</tr>
<tr>
<td>DWG NO</td>
<td></td>
</tr>
<tr>
<td>REV</td>
<td></td>
</tr>
<tr>
<td>SCALE</td>
<td></td>
</tr>
<tr>
<td>SHEET 1 OF 1</td>
<td></td>
</tr>
</tbody>
</table>
Electronics Support Structure

- Diameter: 6.00
- Diameter: 4.00
- Height: 18.25
- Width: 5.00

CHECKED:
QA:
MFG:
APPROVED:

SIZE
DWG NO
REV

SCALE

SHEET 1 OF 1

PRODUCED BY AN AUTODESK EDUCATIONAL PRODUCT
Wing Bracket - Right Wing

<table>
<thead>
<tr>
<th>DRAWN</th>
<th>CHECKED</th>
<th>TITLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>6/11/2011</td>
<td></td>
<td>Wing Bracket - Right Wing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>QA</th>
<th>MFG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>APPROVED</th>
<th>SIZE</th>
<th>DWG NO</th>
<th>REV</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| SCALE       | SHEET 1 OF 1 |
|-------------|---------------|-------------|-----|
Tail Mount

<table>
<thead>
<tr>
<th>DRAWN</th>
<th>6/11/2011</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHECKED</td>
<td></td>
</tr>
<tr>
<td>QA</td>
<td></td>
</tr>
<tr>
<td>MFG</td>
<td></td>
</tr>
<tr>
<td>APPROVED</td>
<td></td>
</tr>
<tr>
<td>TITLE</td>
<td>Tail Mount</td>
</tr>
<tr>
<td>SIZE</td>
<td></td>
</tr>
<tr>
<td>DWG NO</td>
<td></td>
</tr>
<tr>
<td>REV</td>
<td></td>
</tr>
<tr>
<td>SCALE</td>
<td></td>
</tr>
</tbody>
</table>

PRODUCED BY AN AUTODESK EDUCATIONAL PRODUCT
Appendix C

Emergency System C code

This C code is for an Emergency System for an Autonomous Underwater Glider. This system will turn a servo motor to release a drop weight if the temperature, humidity, pressure, and dive time is out of range. An external interrupt is used to receive a high signal from the central computer if there is a problem. After dropping the drop weight, the system will gather GPS coordinates and send them through RF to the nearby boat for glider location. Resources [41], [42], [27], [29] and [36] were used during the programming of the sensor board.

//******************************************************************************
Author: Cheryl Skibski
M.S. Ocean Engineering
Florida Institute of Technology

Resources:
PIC18f45k80 Datasheet
MC18 Compiler Datasheet with MPLAB
SHT15 Sensor Datasheet and Example Code from Sensiron at www.sensirion.com
Sparkfun Electronics
#include <p18lf45k80.h>
#include <delays.h>  //adds the delays library
#include <timers.h>
#include <usart.h>   //adds hardware serial support
#include <stdlib.h>  //adds data conversion library
#include <stdio.h>
#include <string.h>  //adds string tools
#include <math.h>
#include <spi.h>
#include <sw_uart.h> //Software Serial

#define SDA LATBbits.LATB2
#define SCK LATBbits.LATB3

#define noACK 0
#define ACK 1
#define MEASURE_TEMP 0x03 //000 0001 1
#define MEASURE_HUMI 0x05 //000 0010 1

#define Global Variables
/*****************************/
float Date, Time, Lat, Long;
int count=0, min=0, hr=0;
unsigned char check, checksum;
char output[20]="\0";
char *ou=&output;
char output2[20]="\0";
char *ou2=&output2;
float p, pinit;
float tmpcinit, tmpfinit, huminit;
float tempc, tempf, humid;
double gpsdate, gpslat, gpslong, gpstime;
int x=0;
char error="Error has occured. ABORT";
/*****************************/
Define Functions
/*****************************/
void write_sda(void);
void read_sda(void);
void TransmissionStart(void);
char write(unsigned char value);
char read(unsigned char ack);
void ConnectionReset(void);
char measure(unsigned char* sensor_value, unsigned char *sensor_checksum, unsigned char addr);
float pressure(void);
float temphum(float *tmpc, float *tmpf, float *humi);
void emergency(void);
void end(void);

//Define External Interrupt
#pragma code high_vector = 0x08
void high(void)
{
  _asm
  GOTO end
  _endasm
}
#pragma code

/************************************************************
Delay Function for UART at 9600
/************************************************************/

void DelayTXBitUART (void)
{
  Delay10TCYx (9.26); // (((((2*clock)/(4*baud))+1)/2)-12)/10;
     return; for 4Mhz crystal
}

void DelayRXBitUART (void)
{
  Delay10TCYx (9.06); // (((((2*clock)/(4*baud))+1)/2)-14)/10;
     return; 4Mhz crystal
}

void DelayRXHalfBitUART(void)
{
  Delay10TCYx (4.36); // (((((2*clock)/(8*baud))+1)/2)-9)/10;
     return; 4MHz crystal
}
void main(void)
{
    //------Initialization of SHT15 Sensor----------//
    PORTBbits.RB2=0;
    TRISBbits.TRISB2=1;
    TRISBbits.TRISB3 = 0;
    LATBbits.LATB3=0;

    //-------Configure ADC for Pressure Sensor-------//
    TRISAbits.TRISA0=1;
    ADCON0=0b00000000; // select channel AN0, adc idle, enabled
    ADCON1=0b00110000; //Uses the internal vref+ AS 4.096 volts
    ADCON2=0b10001100;

    //-------INT0 External Interrupt----------------//
    TRISBbits.TRISB0=1; // input to pic for external interrupt
    INTCON2bits.INTEDGO=1; //interrupt is triggered on rising edge.
    INTCONbits.INT0IF=0; //clear interrupt flag for int1
    INTCONbits.INT0IE=1; //Enable INT1 External Interrupt
    INTCONbits.GIE = 1; //Enable global interrupts
    ANCON1bits.ANS0=0; //digital pin

    //-------Configure Software UART for Dataloger----//
    OpenUART();

    //-------Configure Hardware Serial Port for GPS at 4800 bps
Open1USART(USART_TX_INT_OFF &
USART_RX_INT_OFF &
USART_ASYNC_MODE &
USART_EIGHT_BIT &
USART_CONT_RX &
USART_BRGH_HIGH,
51.08);

//------Configure Hardware Serial Port for RF at 9600 bps
Open2USART(USART_TX_INT_OFF &
USART_RX_INT_OFF &
USART_ASYNC_MODE &
USART_EIGHT_BIT &
USART_CONT_RX &
USART_BRGH_HIGH,
25.04);

ConnectionReset(); //Resets SHT15 Sensor

//------Take Initial Readings of all sensors -prints integers
temphum(&tmpcinit, &tmpfinit, &huminit);
pinit=pressure();
sprintf(output2, "Initial: %d %d %d %d", (int) tmpcinit, (int) tmpfinit, (int) huminit, (int) pinit);
putsUART(output2);
putcUART(\r); //carriage return
putcUART(\n'); //new line

while(1)
{
count++;

if (count>=72)
{
count=0;
min++;
}
if (min>=72)
{
min=0;
hr++;
}
temphum(&tempc, &tempf, &humid);
p=pressure();

sprintf(output, "%d:%d %d %d %d %d", hr, min, (int) tempc, (int) tempf, (int) humid, (int) p);
putsUART(output);
putcUART(‘\r’); //carriage return
putcUART(‘\n’); //new line

if (tempc>(tmpcinit+5)|humid>(huminit+5)|p>(pinit+5)|min>=20)
{
    putsUART(error);
    emergency();
}

/***********************FUNCTIONS*******/

/*******************************************************************************/
//Emergency Response
/*******************************************************************************/

void emergency(void)
{
    int i, j=0;
    char c;
    char data[70] ="\0";
    char *split;
    const char char delim[] = ",";
    LATD=0b00000010;

    while(j<=3000)
    {
        LATDbits.LATD0=1; // High Signal
        Delay10TCYx(10); // Turn Servo 90 degrees
        LATDbits.LATD0=0; // Low Signal
        j++;
    }
    for (i=0;i<4000;i++) // Delay
        LATDbits.LATD0=0; // Low Signal

RCREG=0;
while(1)
{
    LATD=0b00000010;
    while(!PIR1bits.RCIF);
    c=getc1USART();
    if(c=='$')
    {
        gets1USART(data,70);
    }
    {
        //Parse the GPS String //strtok, strtok_r - split string into //tokens
        split=strtok(data,delim);
        split=strtok(NULL,delim);
        putsUART(split); //Send to Datalogger
        putcUART(0x20);
        puts2USART(split); //Send RF Modem
        putc2USART(0x20);
        Time=atof(split);
        split=strtok(NULL,delim);
        split=strtok(NULL,delim);
        putsUART(split);
        putcUART(0x20);
        puts2USART(split);
        putc2USART(0x20);
        Lat=atof(split);
        split=strtok(NULL,delim);
        split=strtok(NULL,delim);
        putsUART(split);
        putcUART(0x20);
        puts2USART(split);
        putc2USART(0x20);
        Long=atof(split);
        split=strtok(NULL,delim);
        split=strtok(NULL,delim);
        split=strtok(NULL,delim);
        split=strtok(NULL,delim);
        putsUART(split);
        putcUART(0x20);
        puts2USART(split);
        putc2USART(0x20);
        puts2USART(split);
        putc2USART(0x20);
Date=atof(split);
putcUART('
');  // new line
putcUART('');  // carriage return
putc2USART('
');  // new line
putc2USART('');
}
PIR1bits.RCIF=0;  // Clear the interrupt flag
if(RCSTAbits.OERR)  // If we missed a character
{
  RCSTAbits.CREN = 0;  // Reset the error bit
  RCSTAbits.CREN = 1;
}

/************************************************************
Measure Pressure
/************************************************************/
float pressure(void)
{
  unsigned int adresult=0;
  unsigned char *pwork;
  float v, vcor;
  float t;
  ADCON0=0b00000001;  // AN0
  ADCON0bits.GO=1;
  while(ADCON0bits.GO);
  pwork=(unsigned char *)&adresult;
  *pwork=ADRESL;
  pwork=pwork+1;
  *pwork=ADRESH;
  v=adresult*(4.096/4096.00);
  vcor=(v)-0.06;  // Correct Offset due to different voltages
  p=(((vcor/5.0)-0.04)/0.000901)-6;
  return p;
}

/************************************************************
Measure Temperature and Humidity - SHT15 Sensor
/************************************************************/
float temphum(float *tmpc, float *tmpf, float *humi)
{

unsigned char check, checksum;
unsigned int humiint, tempint;
float tempfloat, humifloat, rh_lin;

check=0;
check+=measure((unsigned char*) &tempint, &checksum, MEASURE_TEMP); //measure temperature
check+=measure((unsigned char*) &humiint, &checksum, MEASURE_HUMI); //measure humidity
if (check!=0)
{
    ConnectionReset(); //in case of an check: connection reset
}
else
{

    //--------------Calculate Temperature and Humidity--------//
    tempfloat=(float)tempint; //converts integer to float
    *tmpc=-39.65+(0.01*tempfloat);
    *tmpf=1.8(*tmpc)+32;

    humifloat=(float)humiint; //converts integer to float
    rh_lin=-2.0468+(0.0367*humifloat)+(-1.5955E-6*
        humifloat*humifloat);
    *humi=(*tmpc-25)*(0.01+0.00008*humifloat)+rh_lin;

    /*****************************************************************************/
    Read and Write Functions
    /*****************************************************************************/
    void write_sda(void)
    {
        TRISBbits.TRISB2 = 0; //Set data (B2) as an output
    }

    void read_sda(void)
    {
        TRISBbits.TRISB2 = 1; //Set data (B2) as an input
    }

    /*****************************************************************************/
    Transmission Start sequence: consists of a lowering of the
DATA line
while SCK is high, followed by a low pulse on SCK and
raising DATA again while SCK is still high

void TransmissionStart(void)
{
    write_sda();
    SDA = 1; SCK=0; //data high, clock low
    Delay10TCYx(2);
    SCK = 1;
    Delay10TCYx(2);
    SDA = 0; //Low SDA line while SCK is high
    Delay10TCYx(2);
    SCK = 0; //Low Pulse of SCK
    Delay10TCYx(2);
    SCK = 1; //SCK high
    SDA = 1; //Rise data while SCK is still high
    SCK = 0;
}

char write(unsigned char value)
{
    unsigned char i, check=0;
    write_sda();
    for (i=0x80;i>0;i/=2)
    {
        if (i & value)
            SDA=1; //write to sensor
        else
            SDA=0;
        Delay10TCYx(2);
        SCK=1;
        Delay10TCYx(2);
        Delay10TCYx(2);
        SCK=0;
    }
}
Delay10TCYx(2);
}
SDA=1; //release DATA-line
Delay10TCYx(2);
SCK=1;
check=PORTBbits.RB2; //check ack (DATA will be pulled
//down by SHT15)
SCK=0;
return check; //check=1 in case of no acknowledge
}

/***********************************************************
SHT15 Read - Reads from the SHT15 Sensor
/***********************************************************/
char read(unsigned char ack)
{
unsigned char i,value=0;
SDA=1;
read_sda();
for (i=0x80;i>0;i/=2)
{
  SCK=1;
  Delay100TCYx(100);
  if (PORTBbits.RB2==1)
  {
    value=(value | i);
  }
  SCK=0;
}
write_sda();
Delay10TCYx(2);
SDA=!ack;
SCK=1;
Delay10TCYx(2);
Delay10TCYx(2);
Delay10TCYx(2);
SCK=0;
Delay10TCYx(2);
SDA=1;
return value;
}
SHT15 Reset Interface
/***************************************************************************/
void ConnectionReset(void)
{
    unsigned char i;
    write_sda();
    SDA=1;
    SCK=0;
    for(i=0;i<13;i++)
    {
        SCK=1;
        SCK=0;
    }
    TransmissionStart();
    //Connection reset must be followed by the transmissionstart
}
/***************************************************************************/
SHT15 Measure Sensor
/***************************************************************************/
char measure(unsigned char* sensor_value, unsigned char *sensor_checksum, unsigned char addr)
{
    unsigned char check=0;
    unsigned long i;
    TransmissionStart(); //transmission start
    check+=write(addr);
    for (i=0;i<0xFFFFFF;i++)
    {
        if(PORTBbits.RB2==0)
        {
            break; //wait until sensor has finished the measurement
        }
    }
    if(PORTBbits.RB2) check+=1;
    *(sensor_value+1)=read(ACK); //read the first byte (MSB)
    *(sensor_value)=read(ACK); //read the second byte (LSB)
    *sensor_checksum =read(noACK); //read checksum
    return check;
}

#pragma interrupt end
void end(void)
{ 

if(INTCONbits.INT0IF&& INTCONbits.INT0IE) 
{
emergency();
}
}

/**************************END**************************/

/**************************END**************************/

Additional Timer Function not implemented in cod above

/**************************END**************************/

//void timer(void)
{
///// Configure Timer1
// T1CON=0b00000000; //Prescalor of 1:1
// PIR1bits.TMR1IF=0; //Clear the interrupt flag
// T1CONbits.TMR1ON=1; //Start Timer1
// PIE1bits.TMR1IE=1; //Enable overflow interrupt
// const int htmr=0xF8;
// const int ltmr=0x30;
//
// if(PIE1bits.TMR1IE)
// {
// ms++;
//}
// if(ms>=30)
// {
// s++;
// ms=0;
// }
// if(s>=60)
// {
// s=0;
// min++;
// }
// if(min>=60)
// {
// min=0;
// hr++;
// }
//
// T1CONbits.TMR1ON=0; //reload timer 1 for frequency
// TMR1H=htmr;
// TMR1L=ltmr;
// T1CONbits.TMR1ON=1;
// PIR1bits.TMR1IF=0;  //Clear overflow flag for next time
// }
//}
Appendix D

Emergency System Schematic

The following schematics include the emergency system sensor and the RF receiver board.
RF Receiving Circuit

TITLE: RFBoard

Document Number:  

REV:

Date: 8/7/2011 3:17:38 PM  
Sheet: 1/1
Appendix E

Gumstix Computer and Sensor Schematic

The following schematics are for the Tobi Expansion board 40-pin header and the logic level converter from Sparkfun Electronics.
Appendix F

Overo Fire Python Code

The following python code is used to demonstrate the vehicle control using wing control flaps. Compass data is recorded and the flaps change depending on roll. Resources [34], [26], [29], [24], and [25] were used while programming for the Overo Fire. This code has a timer that runs and at the end of the timer, the GPIO signal is set to high and sent to the emergency system to drop the weight.

```python
def pwm10init():
    os.system("devmem2 "+pwm10addr[0:7]+"024 w 0")
    os.system("devmem2 0x4800"+muxoffset10+" h "+muxval10)
    os.system("devmem2 "+pwm10addr[0:7]+"02C w 0xFFFFFD80")

def pwm10generate(tmar):
    os.system("devmem2 "+pwm10addr[0:7]+"038 w "+tmar)
    os.system("devmem2 "+pwm10addr[0:7]+"028 w 0xFFFFFD80")
    os.system("devmem2 "+pwm10addr[0:7]+"024 w 0x01843")
    print "Outputting PWM for Left Wing"

def pwm11init():
    os.system("devmem2 "+pwm11addr[0:7]+"024 w 0")
    os.system("devmem2 0x4800"+muxoffset11+" h "+muxval11)
    os.system("devmem2 "+pwm11addr[0:7]+"02C w 0xFFFFFD80")
```

154
def pwm11generate(tmarval):
    os.system("devmem2 " + pwm11addr[0:7] + "038 w " + str(tmarval))
    os.system("devmem2 " + pwm11addr[0:7] + "028 w 0xFFFFFD80")
    os.system("devmem2 " + pwm11addr[0:7] + "024 w 0x01843")
    print "Outputting PWM for Right Wing"

def leftup20():
pwm10generate(twenty)
sleep(2.5)
pwm10generate(zero)

def leftdown20():
pwm10generate(ntwenty)
sleep(2.5)
pwm10generate(zero)

def rightup20():
pwm11generate(ntwenty)
sleep(2.5)
pwm11generate(twenty)

def rightdown20():
pwm11generate(twenty)
sleep(2.5)
pwm11generate(ntwenty)

def wingleft():
pwm10generate(fifteen) #15
sleep(2.5)
pwm10generate(zero)
sleep(2.5)
pwm10generate(nfifteen) #15 opposite
sleep(2.5)
pwm10generate(zero)
sleep(2.5)
pwm10generate(twenty) #20
sleep(2.5)
pwm10generate(zero)
sleep(2.5)
pwm10generate(ntwenty) #20 opposite
sleep(2.5)
pwm10generate(zero)
sleep(2.5)
pwm10generate(fifteen)
sleep(2.5)
pwm10generate(zero)
sleep(2.5)
pwm10generate(nfifteen)
sleep(2.5)
pwm10generate(zero)
sleep(2.5)
pwm10generate(twenty)
sleep(2.5)
pwm10generate(zero)
sleep(2.5)
pwm10generate(ntwenty)
sleep(2.5)
pwm10generate(zero)

def wingright():
pwm11generate(nfifteen)
sleep(2.5)
pwm11generate(zero)
sleep(2.5)
pwm11generate(fifteen)
sleep(2.5)
pwm11generate(zero)
sleep(2.5)
pwm11generate(ntwenty)
sleep(2.5)
pwm11generate(zero)
sleep(2.5)
pwm11generate(twenty)
sleep(2.5)
pwm11generate(zero)
sleep(5)
pwm11generate(fifteen)
sleep(2.5)
pwm11generate(zero)
sleep(2.5)
pwm11generate(ntwenty)
sleep(2.5)
pwm11generate(zero)
sleep(2.5)
pwm11generate(twenty)
sleep(2.5)

pwm11generate(zero)
sleep(2.5)
pwm11generate(twenty)
sleep(2.5)
pwm11generate(zero)

def gps():
gpsdata=[]
gpsdir=[]
s=serial.Serial('/dev/ttyS2', 4800, timeout=1)
while 1:
gps=s.readline()
if gps.startswith("$GPRMC,"):
    s.close()
gps=gps.split("\","")
Time=gps[1]
Latitude=gps[3]
LatDir=gps[4]
Longitude=gps[5]
LongDir=gps[6]
Speed=gps[7]
Course=gps[8]
Date=gps[9]
gpsdir.append(LatDir)
gpsdir.append(LongDir)
gpsdata.append(Time)
gpsdata.append(Date)
gpsdata.append(Latitude)
gpsdata.append(Longitude)
gpsdata.append(Speed)
gpsdata.append(Course)
return gpsdata

def imu():
    AccelX=None
    AccelY=None
    AccelZ=None
    Pitch=None
    Yaw=None
    Roll=None
    imudata=[]
s=serial.Serial('/dev/ttyUSB0', 115200)
imu=s.readline()
s.close()
data=imu.split()
count=data[1]
AccelX=data[2]
AccelY=data[3]
AccelZ=data[4]
Pitch=data[5]
Roll=data[6]
Yaw=data[7]
AccelXV=(float(AccelX)*(3.7/1023)-1.65)/0.2
AccelYV=(float(AccelY)*(3.7/1023)-1.65)/0.2
AccelZV=(float(AccelZ)*(3.7/1023)-1.65)/0.2
PitchV=(float(Pitch)*(3.7/1023)-1.65)/.0033
RollV=(float(Roll)*(3.7/1023)-1.65)/.0033
YawV=(float(Yaw)*(3.7/1023)-1.65)/.0033
imudata.append(AccelXV)
imudata.append(AccelYV)
imudata.append(AccelZV)
imudata.append(PitchV)
imudata.append(RollV)
imudata.append(YawV)
return imudata

def compass():
compassdata=[]
s=serial.Serial('/dev/ttyS0', 9600, timeout=1)
s.write('#')
compass=s.read(4)
#s.close()
heading=None
pitch=None
roll=None
if(len(compass)==4):
pitch=ord(compass[2])
pitch=-1*pitch
roll=ord(compass[3]):-1
heading=ord(compass[0])*256+ord(compass[1])
if abs(roll)>90:
roll=roll+255
if abs(pitch)>90:
pitch=pitch+255
compassdata.append(heading)
compassdata.append(pitch)
compassdata.append(roll)
return compassdata

def gpioinit():
gpio="/sys/class/gpio/
gpioexport=open(gpio+"export", "w")
gpioexport.write("147")
#gpioexport.close()

def gpio():
gpio147="/sys/class/gpio/gpio147/
gpiodir=open(gpio147+"direction", "w")
gpiodir.write("out")
#gpiodir.close()
gpioval=open(gpio147+"value", "w")
gpioval.write("1")
print "GPIO147 is HIGH"
#gpioval.close()

###--MAIN---##
import os
from time import localtime, clock, sleep
import threading
from threading import Thread
import serial

filename="/home/root/internalMemory/data.txt"
datafile=open(filename, "w")

gpsdata=[]
imudata=[]
data=[]
compassdata=[]

##--Initalize variables--#
pwm10addr="0x48086000"
pwm11addr="0x48088000"

##--TMAR values for each angle--#
five="0xFFFFFD00" #left wing down, right wing up
nfive="0xFFFFFDB2" #left wing up, right wing down
ten="0xFFFFFDAB"
nten="0xFFFFFDAB"
fifteen="0xFFFFFDB3"
nfifteen="0xFFFFFDAA"
twenty="0xFFFFFDB5"
ntwenty="0xFFFFFDA8"
zero="0xFFFFFDAA"

##--Mux values for pins--##
muxoffset10="2176"
muxval10="0x0002"
muxoffset11="2178"
muxval11="0x00102"

##--initialze pwm and gpio--##
gpioinit()
pwm10init()
pwm11init()

##--Call function to generate signals------##
#Use this block only for wing flap control test
#Uncomment the last two lines and just run this
##The two functions will run at the same time
#To test movement of the flaps
leftwing=threading.Thread(target=wingleft)
rightwing=threading.Thread(target=wingright)
leftwing.daemon=True
rightwing.daemon=True
leftwing.start() #To test wings
rightwing.start() #To test wings

gpsdata=gps()
print gpsdata

sleep(2) #

#--Get initial Data---#
initcompass=[]
initcompass=compass()
print initcompass

eta=0.10
start=clock()

pwm10generate(twenty)
pwm11generate(ntwenty)
sleep(2.5)

while 1:
    compassdata=compass()
data=compassdata
    sleep(2)  #Read every 2 seconds data
    print data

    ###---While Diving State----###
    if data[2]>initcompass[2]:
        print "Rolling to Left"
pwm10generate(twenty)
pwm11generate(twenty)
sleep(2.5)
        print "Rolling to Right"
pwm11generate(ntwenty)
pwm10generate(ntwenty)
sleep(2.5)
    else:
        print "Not Rolling"

    time=clock()-start
    print time

    if time>=eta:
        pwm10generate(zero)
pwm11generate(zero)
sleep(5)
        print "At Bottom"
        gpio()
pwm10generate(20)
pwm11generate(20)

    for item in data:
datafile.write(str(item)+",")
Appendix G

PWM Register Calculations

This section includes the calculation of solution and steps of servo rotation, as well as the calculation of hex values needed to write to the registers to output the PWM signal.

```python
import sys
from math import *

#Known Values
FCLK=32000  #Clock Frequency of PWM pin
DutyCycle=0.076  #Duty Cycle in Decimal (percentage/100)
Freq=50       #Frequency in Hz Period is 20 ms

#Calculated Values
TLDR=4294967295-(FCLK/Freq)+1
NUM_Settings=4294967294-TLDR
TMAR=TLDR+(NUM_Settings*DutyCycle)

#Print Values
print "Operating Frequency " +str(FCLK)
print "Frequency " +str(Freq)
print "Duty Cycle " +str(DutyCycle)
print "Num Settings " +str(NUM_Settings)
print "TLDR Value " +str(TLDR)
```
print "TMAR Value " +str(floor(TMAR))

#Calculation of Resolution and Steps of servo rotation
duty1=6.4
duty2=8.4
pulsehighms=1650
pulselowms=1250
dutyrange=duty2-duty1
steps=(dutyrange/100)*NUM_Settings
pulserange=pulsehighms-pulselowms
resolution=pulserange/steps

print "Duty Cycle Range " +str(dutyrange)
print "Number of Steps for Duty Range " +str(steps)
print "Pulse Width Range " +str(pulserange)
print "Resolution " +str(resolution)
Appendix H

Cost List

The following cost list includes the projected cost for this glider.

<table>
<thead>
<tr>
<th>Item</th>
<th>Total Price</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fairing</td>
<td>$120.00</td>
<td>Fiberglass Florida</td>
</tr>
<tr>
<td>Aluminum 6061-T6</td>
<td>$200.00</td>
<td>Alro</td>
</tr>
<tr>
<td>O-ring</td>
<td>$2.36</td>
<td>Parker O-Ring</td>
</tr>
<tr>
<td>Polyethylene</td>
<td>$12.74</td>
<td>McMaster Carr</td>
</tr>
<tr>
<td>threaded rods</td>
<td>$23.52</td>
<td>Home Depot</td>
</tr>
<tr>
<td>hardware</td>
<td>$5.90</td>
<td>Home Depot</td>
</tr>
<tr>
<td>1-GumStix Overo Fire</td>
<td>$219.00</td>
<td>Gumstix</td>
</tr>
<tr>
<td>1-Tobi Expansion Board</td>
<td>$69.00</td>
<td>Gumstix</td>
</tr>
<tr>
<td>2-ufl to SMA Antenna Connecters</td>
<td>$9.80</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>1-Tilt Compensated Compass</td>
<td>$71.96</td>
<td>Robotshop</td>
</tr>
<tr>
<td>1-PMB-688 SiRF GPS</td>
<td>$39.99</td>
<td>Parallax Inc</td>
</tr>
<tr>
<td>1-GPS Antenna Extension</td>
<td>$9.99</td>
<td>Parallax Inc</td>
</tr>
<tr>
<td>1-FTDI Basic Breakout</td>
<td>$14.95</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>1-Six DOF IMU</td>
<td>$79.95</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>1-LiPo Battery</td>
<td>$24.95</td>
<td>Sparkfun</td>
</tr>
</tbody>
</table>

continued on next page

164
<table>
<thead>
<tr>
<th>Item</th>
<th>Total Price</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-5.0 voltage regulator</td>
<td>$1.25</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>1-3.3 voltage regulator</td>
<td>$1.25</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>2-LEDs</td>
<td>$0.70</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>2-logic level converters</td>
<td>$3.90</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>5-Resistors</td>
<td>$1.25</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>2-Capacitors</td>
<td>$0.12</td>
<td>Jameco</td>
</tr>
<tr>
<td>1-LM339 Comparator</td>
<td>$0.25</td>
<td>Jameco</td>
</tr>
<tr>
<td>2-Servo Motors</td>
<td>$33.90</td>
<td>Jameco</td>
</tr>
<tr>
<td>1-USB Standard to mini-B cable</td>
<td>$2.73</td>
<td>Amazon</td>
</tr>
<tr>
<td>1-Powered USB Hub</td>
<td>$23.13</td>
<td>Amazon</td>
</tr>
<tr>
<td>1-5.0 volt power adapter</td>
<td>$10.00</td>
<td>Gumstix</td>
</tr>
<tr>
<td>1-USB Cable mini-B to mini-A</td>
<td>$8.00</td>
<td>Gumstix</td>
</tr>
<tr>
<td>1-HDMI to DVI Cable</td>
<td>$15.00</td>
<td>Gumstix</td>
</tr>
<tr>
<td>2-Adaptabplug Socket</td>
<td>$9.98</td>
<td>Radioshack</td>
</tr>
<tr>
<td>2-Adaptabplug Tip</td>
<td>$13.98</td>
<td>Radioshack</td>
</tr>
<tr>
<td>3-9.0 volt batteries</td>
<td>$6.99</td>
<td>Amazon</td>
</tr>
<tr>
<td>1-SHT15 Humidity Temperature</td>
<td>$41.95</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>1-Pressure Sensor</td>
<td>$34.95</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>1-PIC18LF45K80</td>
<td>$3.18</td>
<td>Microchip</td>
</tr>
<tr>
<td>1-GPS</td>
<td>$39.99</td>
<td>Parallax Inc</td>
</tr>
<tr>
<td>2-RF Xbee Pro</td>
<td>$179.80</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>2-Xbee Breakout boards and pins</td>
<td>$19.80</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>1-Servo Motor</td>
<td>$16.95</td>
<td>Jameco</td>
</tr>
<tr>
<td>1-Datalogger</td>
<td>$24.95</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>1-FTDI Basic Breakout</td>
<td>$14.95</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>1-USB Standard to mini-B cable</td>
<td>$2.73</td>
<td>Amazon</td>
</tr>
<tr>
<td>1-Crystal</td>
<td>$0.95</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>3-Capacitors</td>
<td>$0.18</td>
<td>Jameco</td>
</tr>
<tr>
<td>2-LED</td>
<td>$0.70</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>3-Resistor</td>
<td>$0.75</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>1-9.0 volt battery</td>
<td>$2.33</td>
<td>Amazon</td>
</tr>
<tr>
<td>1-3.3 volt regulator</td>
<td>$1.25</td>
<td>Sparkfun</td>
</tr>
</tbody>
</table>

*continued on next page*
<table>
<thead>
<tr>
<th>Item</th>
<th>Total Price</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-5.0 volt regulator Fiberglass Resin and Catalyst</td>
<td>$1.25</td>
<td>Sparkfun</td>
</tr>
<tr>
<td>2-Rods</td>
<td>$3.98</td>
<td>Structural Composites</td>
</tr>
<tr>
<td>1-Control Horn</td>
<td>$5.95</td>
<td>Hobbytown USA</td>
</tr>
<tr>
<td>1-Pack of Servo Arms</td>
<td>$15.95</td>
<td>Hobbytown USA</td>
</tr>
<tr>
<td>1-Threaded Inserts</td>
<td>$22.55</td>
<td>McMastcarr</td>
</tr>
<tr>
<td>1-Stainless bolts</td>
<td>$3.54</td>
<td>Home Depot</td>
</tr>
<tr>
<td>1-HDPE material</td>
<td>$12.74</td>
<td>McMastcarr</td>
</tr>
<tr>
<td><strong>Total Cost</strong></td>
<td><strong>$1,489.61</strong></td>
<td></td>
</tr>
</tbody>
</table>
Appendix I

Datasheets

The following are datasheets for sensors and the Gumstix Overo Fire processor.
The HS-311 sport servo provides the novice modeler all the performance and reliability you would expect to find in a more expensive servo. Combined with precise resin gear and SMT circuitry, the HS-311 represents a remarkable value in today's R/C market.

**Detailed Specifications**

- **Motor Type:** 3 Pole
- **Steering Type:** Nylon
- **Speed:** 0.19 / 0.15 sec @ 50 deg
- **Torque:** 42 / 48.0 oz.in (4.8 / 5.6 N.m)
- **Size:** 1.57" x 0.98" x 1.43"
- **Weight:** 1.51 oz

**Available Models**

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HS-311S</td>
<td>HS-311</td>
</tr>
</tbody>
</table>

**Spare Parts**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Servo Case Set</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Servo Gear Set</strong></td>
<td>Nylon &amp; Carbonite</td>
</tr>
<tr>
<td><strong>Servo Horn &amp; Hardware Set</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Detailed Feature Descriptions**

---

168
Overview

Need a jump start on your robotics project, but strapped for cash? Is your junk food budget cutting into your autonomous monkey-shaving-robot plans? Is your quest for world domination constantly thwarted, week after week, because you always end up 25 bucks short of a 6DOF v4?

The 6DOF Atomic is a stripped-down IMU unit, designed to give good performance at a low price. The unit can run as a hard-wired UART interface (0-3.3V, 115200bps), or optionally with an XBee TM RF module, and is powered from a single LiPo (Lithium Polymer) cell.

The processor is an Atmel ATMega168TM running at 10MHz with 6 dedicated 10-bit ADC channels reading the sensors. Source code for the 6DOF Atomic is freely available and compiles with the free AVR GCC compiler.

The 6-DOF Atomic uses these sensors:

- Freescale MMA7260Q TM triple-axis accelerometer, settable to 1.5g, 2g, 4g or 6g sensitivity
- 3 ST Microelectronics LISY300AL TM single-axis, 300°/s gyros

Electrical Specifications:

- Input voltage: 3.4V to 10V DC
- Current consumption: 24mA (75mA with XBee)
- Sensor bandwidth and resolution:
  - LISY300AL Gyros: 88Hz, 0.977°/tick (ADC count)
  - MMA7260Q Accelerometer:
    - 350Hz, X and Y axes
    - 150Hz, Z axis
    - 0.00403g/tick @ 1.5g
    - 0.00537g/tick @ 2g
    - 0.0107g/tick @ 4g
    - 0.0161g/tick @ 6g

The MMA7260Q accelerometer and the LISY300AL gyros have each been set up per their manufacturer’s recommendations, i.e. internal clock suppression filters on their outputs.

These sensors are also internally temperature compensated. For a full description of the sensor specifications, please see the respective manufacturer’s data sheets (available at http://www.sparkfun.com/).

Hardware Overview

1) Power indicator LED
2) Reset switch for the ATMega168
3) Power Switch
4) JST power connector for single Lipo cell
5) Serial port, 0-3.3V, 115200/8/1/N. The unit can optionally be powered with 3.4V to 10V from this port by closing the solder jumper marked “9” in Figure 1.
6) Programming port for ATMega168
Atomic IMU - 6 Degrees of Freedom - XBee Ready
2009.3.25

7) Status LED, blinks at the sample rate when unit is sampling, off otherwise
8) Sockets for optional XBee RF module
9) Solder jumper, connects VCC on the serial port to the battery input

Setup

When you first power up the 6DOF Atomic, you will see the power indicator LED light up and status LED blink 5 times quickly. During normal sampling operations, the status LED will toggle on or off every 64 sampling cycles. When in the configuration menu, the status LED will be on continuously. If the device is not in the configuration menu and not sampling, the 6DOF Atomic is in its idle state and the status LED will be off (see the section on the configuration menu auto run mode for more details on this).

The 6DOF Atomic can run either as a hard-wired device or optionally with a pair of XBee RF modules. The unit uses the same UART for both operations so no firmware configuration is required to run in either mode. However, care must be taken not to connect both a hard line and an XBee at the same time as that will result in a UART conflict and may possibly damage the unit.

Hard line connection

If the user chooses a hard-wired connection to the Atomic, the serial UART lines can be accessed on the serial port header (marked “5” in Figure 1). TX, RX and ground are all that’s required. The logic is 0 – 3.3V, but the lines are 5V tolerant. DO NOT connect RS-232 to the UART lines as you will most likely damage the unit.

The serial port header is lined up to match the SparkFun FTDI Basic Breakout – 5V (sku: DEV-09115). It provides the quickest way to get up and running with the 6DOF Atomic. The user only needs to install a right-angled male header into the serial port and plug them together.

XBee connection

To establish a connection over XBee modules, the user must first purchase the necessary hardware, such as two XBee 1mW Chip Antenna modules and an XBee Explorer USB (sku: WRL-08664 and WRL-08687). Then you only need to follow the beginning part of the Wireless XBee/AVR Bootloading tutorial (setting up the XBee modules, http://www.sparkfun.com/commerce/tutorial_info.php?tutorials_id=122), substituting 115200 baud for 19200 baud. Once the setup is complete, the XBee modules will operate transparently.

It is worth noting, however, that stopping the 6DOF Atomic over the XBee link at frequencies over 150Hz can be problematic. With so much data streaming over the link it becomes difficult to get the stop signal back to the 6DOF. The only way to stop in this situation is to hit the reset button or cycle power.

6DOF Atomic Mixer demo application

For a demonstration of the 6DOF Atomic’s operation, the user can run the 6DOF Atomic Mixer program. With the 6DOF powered up and connected in its idle state, close any terminal programs open the 6DOF’s serial port and start the mixer application. Select the port number to which the 6DOF (or XBee base unit) is connected, set the frequency and sensitivity, hit the start button and you’re off and running.

The mixer program requires that all channels are active. If you have any trouble getting the program to work correctly, check that all channels are active in the configuration menu.

Using a Terminal and the Configuration Menu

Once the initial novelty of the mixer program wears off, the user may want to do something slightly more useful with the Atomic, like attach it to something and log a file with a terminal program. What follows is a brief description of the configuration menu as well as a functional description of the 6DOF Atomic’s operation.

Operation from the Idle State

Upon reset and in its default configuration, the Atomic will check to see if it has been configured for “auto run” mode (more on that later). If it’s not in auto run mode, it goes into an idle state waiting for input. Normally this idle state serves as the start up state for the mixer application. In this state, the following inputs have the following effects:

1) “%”, ASCII 37, sets the accelerometer to 1.5g sensitivity
2) “&”, ASCII 38, sets the accelerometer sensitivity to 2g
3) " ' " (apostrophe), ASCII 39, sets the accelerometer sensitivity to 4g
4) " ( ), ASCII 40, sets the accelerometer sensitivity to 6g
5) " ) " ASCII 41, sets the sample frequency to 50Hz
6) " * " ASCII 42, sets the sample frequency to 100Hz
7) " + " ASCII 43, sets the sample frequency to 150Hz
8) " , " ASCII 44, sets the sample frequency to 200Hz
9) " - " ASCII 45, sets the sample frequency to 250Hz
10) " # " ASCII 35, starts the unit running in binary mode with all channels active
11) " " (space), ASCII 32, stops the unit and returns it to the idle state (issuing another ASCII 32 will bring up the configuration menu)

Operation from this idle state will always be in binary output mode, but the user may select which channels are active. Also, configuration from this state as done in the actual configuration menu will be saved to memory for future use. At the same time, all but the active channel settings saved in memory have no effect on operation from the idle state.

It should be noted that the primary purpose of this idle state and mode of operation is to more easily interact with the Atomic mixer demonstration application, but there’s no reason that a user’s own application couldn’t use it for a quick setup.

Operation from the Configuration Menu

6DOF Atomic setup, version 1.0

1) View/edit active channel list
2) Change output mode, currently binary
3) Set Auto run mode, currently off
4) Set accelerometer sensitivity, currently 1.5g
5) Set output frequency, currently 100
6) Save settings and run unit

Active Channel List

Pressing "1" will bring up the active channel list:

1) Accel X = on
2) Accel Y = on
3) Accel Z = on
4) Pitch = on
5) Roll = on
6) Yaw = on

Press the number of the channel you wish to change, or press x to exit

To change a channel from active to inactive (or the reverse), just press the number of the channel you wish to change. It’s a toggling function; pressing a number will bring up the full list again, but with the channel you wished to change in its opposite state. Press a few numbers and get a feel for it. Pressing "x" gets you back to the main menu.

Output Mode

Pressing “2” from the main menu will toggle the output mode from binary to ASCII and back again. What are these output modes, you ask?

In both output modes, the data from all active channels is framed by an “A” (ASCII 65) at the start and a “Z” (ASCII 90) at the end. Also in both modes, each channel is reported in exactly the sequence shown in the active channel list, with the addition of a sample count that immediately follows the “A” and precedes the first active measurement, which is to say:

1) Count
2) Accel X
3) Accel Y
4) Accel Z
5) Pitch
6) Roll
7) Yaw

The count is two bytes that comes as MSB-LSB, and will range from 0 to 32767. If any of the channels are selected as inactive, that data is omitted from the frame and subsequent data moves up in the report sequence.

In binary mode, each active channel report comes as 2 bytes: MSB and LSB, in that sequence, and they will always be between 0 and 1023 because we’re reading from 10-bit ADC’s. The width of the data...
frame in binary mode will be 4 bytes ("A", "Z", and count are always present) plus 2 bytes for each active measurement. So for all active channels the data frame will be 16 bytes wide.

In ASCII mode, the count and active measurements are reported in ASCII so it’s easier to read with a terminal program, plus all measurements and the count are delimited with TAB characters (ASCII 9) as well as a carriage return and line feed at the end of the data frame. This makes data capture and importation into a spreadsheet a relatively simple matter.

Auto Run Mode

Pressing “3” from the main menu will toggle the auto run setting. If you intend to use the 6DOF Atomic in ASCII mode, set this to “on”. If the auto run feature is off, the unit will always run from its primary idle state, which means that it will always wait for a "#" to begin sampling and it will always run in binary mode.

One feature of auto run mode is that if the setting is active the Atomic will begin sampling immediately upon power up. Pressing the spacebar will bring up the configuration menu again.

Setting the Accelerometer Sensitivity

Pressing “4” from the main menu will bring up the following submenu:

Set to:
  1) 1.5g
  2) 2g
  3) 4g
  4) 6g

Just press the number which corresponds to your choice and the unit will revert to the main menu with the sensitivity changed.

Setting the Output Frequency

Pressing “5” from the main menu will allow you to change the sample frequency. Simply press “i” to increase or “d” to decrease, or “x” to revert to the main menu.

The minimum frequency setting is 10Hz, and there is no maximum setting. This allows the user to experiment with smaller data frames and higher sampling rates.

Save Settings and Run Unit

Pressing “9” from the main menu will save the current settings to flash and exit the configuration menu. If the auto run feature has been activated the unit will begin running immediately. If it has not been set, the unit will revert to the initial idle state and wait for additional input.

Bandwidth Considerations and Firmware

The 6DOF Atomic does not have any filtering in firmware, though there is enough memory left in the ATmega168 flash program space to implement filtering. The internally set output bandwidth of the MMA7260Q accelerometer is 350Hz for the X and Y axis, and 150Hz for the Z axis. There are also additional single-poll low-pass filters to reduce switching noise from the sensor with polls set at 1591Hz (recommended by Freescale). The internally set output bandwidth for the LISY300AL gyro sensors is 88Hz. Of course, it’s a good idea for the user to consider these numbers when developing an application to ensure that the proper filtering is in place for whatever sampling rate is selected.

All source code and schematics for the 6DOF Atomic are freely available from the Atomic’s product description page on http://www.sparkfun.com. Source code is compiled with GCC using WinAVR (http://winavr.sourceforge.net/). Programming is typically done with a AVR-PG1B (sku: PGM-00014, $12.95) along with our programming adapter (sku: BOB-08508).
±1.5g - 6g Three Axis Low-g Micromachined Accelerometer

The MMA7260Q low cost capacitive micromachined accelerometer features signal conditioning, a 1-pole low pass filter, temperature compensation and g-Select which allows for the selection among 4 sensitivities. Zero-g offset full scale span and filter cut-off are factory set and require no external devices. Includes a Sleep Mode that makes it ideal for handheld battery powered electronics.

Features
- Selectable Sensitivity (1.5g/2g/4g/6g)
- Low Current Consumption: 500 µA
- Sleep Mode: 3 µA
- Low Voltage Operation: 2.2 V – 3.6 V
- 6mm x 6mm x 1.45mm QFN
- High Sensitivity (800 mV/g @1.5 g)
- Fast Turn On Time
- High Sensitivity (1.5 g)
- Integral Signal Conditioning with Low Pass Filter
- Robust Design, High Shocks Survivability
- Pb-Free Terminations
- Environmentally Preferred Package
- Low Cost

Typical Applications
- HDD MP3 Player : Freefall Detection
- Laptop PC : Freefall Detection, Anti-Theft
- Cell Phone : Image Stability, Text Scroll, Motion Dialing, E-Compass
- Pedometer : Motion Sensing
- PDA : Text Scroll
- Navigation and Dead Reckoning : E-Compass Tilt Compensation
- Gaming : Tilt and Motion Sensing, Event Recorder
- Robotics : Motion Sensing

ORDERING INFORMATION

<table>
<thead>
<tr>
<th>Device Name</th>
<th>Temperature Range</th>
<th>Case No.</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMA7260Q</td>
<td>– 20 to +85°C</td>
<td>1622-01</td>
<td>QFN-16, Tube</td>
</tr>
<tr>
<td>MMA7260QR2</td>
<td>– 20 to +85°C</td>
<td>1622-01</td>
<td>QFN-16, Tube &amp; Reel</td>
</tr>
</tbody>
</table>

© Freescale Semiconductor, Inc., 2005. All rights reserved.
WARNING: This device is sensitive to electrostatic discharge.

Although the Freescale accelerometer contains internal 2000 V ESD protection circuitry, extra precaution must be taken by the user to protect the chip from ESD. A charge of over 2000 volts can accumulate on the human body or associated test equipment. A charge of this magnitude can alter the performance or cause failure of the chip. When handling the accelerometer, proper ESD precautions should be followed to avoid exposing the device to discharges which may be detrimental to its performance.

Table 1. Maximum Ratings

(Maximum ratings are the limits to which the device can be exposed without causing permanent damage.)

<table>
<thead>
<tr>
<th>Rating</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Acceleration (all axis)</td>
<td>$g_{\text{max}}$</td>
<td>±2000 g</td>
<td></td>
</tr>
<tr>
<td>Supply Voltage</td>
<td>$V_{\text{DD}}$</td>
<td>–0.3 to +3.6</td>
<td>V</td>
</tr>
<tr>
<td>Drop Test(1)</td>
<td>$D_{\text{drop}}$</td>
<td>1.8 m</td>
<td></td>
</tr>
<tr>
<td>Storage Temperature Range</td>
<td>$T_{\text{stg}}$</td>
<td>–40 to +125</td>
<td>°C</td>
</tr>
</tbody>
</table>

1. Dropped onto concrete surface from any axis.

ELECTRO STATIC DISCHARGE (ESD)

Table 2. Maximum Ratings
Table 2. Operating Characteristics
Unless otherwise noted: –20°C < \( T_A \) < 85°C, 2.2 V < \( V_{DD} \) < 3.6 V, Acceleration = 0g, Loaded output\(^{(1)}\)

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Symbol</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Range(^{(2)})</td>
<td>( V_{DD} )</td>
<td>2.2</td>
<td>3.3</td>
<td>3.6</td>
<td>V</td>
</tr>
<tr>
<td>Supply Voltage(^{(3)})</td>
<td>( V_{DD} )</td>
<td>—</td>
<td>500</td>
<td>800</td>
<td>( \mu A )</td>
</tr>
<tr>
<td>Supply Current</td>
<td>( I_{DD} )</td>
<td>—</td>
<td>3</td>
<td>10</td>
<td>( \mu A )</td>
</tr>
<tr>
<td>Supply Current at Sleep Mode(^{(4)})</td>
<td>( I_{DD} )</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Operating Temperature Range</td>
<td>( T_A )</td>
<td>–20</td>
<td>—</td>
<td>+85</td>
<td>°C</td>
</tr>
<tr>
<td>Acceleration Range, X-Axis, Y-Axis, Z-Axis</td>
<td>( g_{FS} )</td>
<td>—</td>
<td>±1.5</td>
<td>—</td>
<td>g</td>
</tr>
<tr>
<td></td>
<td>( g_{FS} )</td>
<td>—</td>
<td>±2.0</td>
<td>—</td>
<td>g</td>
</tr>
<tr>
<td></td>
<td>( g_{FS} )</td>
<td>—</td>
<td>±4.0</td>
<td>—</td>
<td>g</td>
</tr>
<tr>
<td></td>
<td>( g_{FS} )</td>
<td>—</td>
<td>±6.0</td>
<td>—</td>
<td>g</td>
</tr>
<tr>
<td>Output Signal</td>
<td>( V_{OFF} )</td>
<td>1.485</td>
<td>1.65</td>
<td>1.815</td>
<td>V</td>
</tr>
<tr>
<td>Zero g (( T_A = 25°C, V_{DD} = 3.3 ) V)</td>
<td>( V_{OFF} )</td>
<td>—</td>
<td>±2</td>
<td>—</td>
<td>mg/°C</td>
</tr>
<tr>
<td>Sensitivity (( T_A = 25°C, V_{DD} = 3.3 ) V)</td>
<td>( S_{1.5g} )</td>
<td>740</td>
<td>800</td>
<td>860</td>
<td>mV/g</td>
</tr>
<tr>
<td></td>
<td>( S_{2g} )</td>
<td>555</td>
<td>600</td>
<td>645</td>
<td>mV/g</td>
</tr>
<tr>
<td></td>
<td>( S_{4g} )</td>
<td>277.5</td>
<td>300</td>
<td>322.5</td>
<td>mV/g</td>
</tr>
<tr>
<td></td>
<td>( S_{6g} )</td>
<td>185</td>
<td>200</td>
<td>215</td>
<td>mV/g</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>( S_{TA} )</td>
<td>—</td>
<td>±0.03</td>
<td>—</td>
<td>%/°C</td>
</tr>
<tr>
<td>Bandwidth Response</td>
<td>( f_{3dB} )</td>
<td>—</td>
<td>350</td>
<td>—</td>
<td>Hz</td>
</tr>
<tr>
<td>XY</td>
<td>( f_{3dB} )</td>
<td>—</td>
<td>150</td>
<td>—</td>
<td>Hz</td>
</tr>
<tr>
<td>Z</td>
<td>( f_{3dB} )</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Noise</td>
<td>( \eta_{RMS} )</td>
<td>—</td>
<td>4.7</td>
<td>—</td>
<td>mVrms</td>
</tr>
<tr>
<td>Power Spectral Density RMS (0.1 Hz – 1 kHz)(^{(4)})</td>
<td>( \eta_{PSD} )</td>
<td>—</td>
<td>350</td>
<td>—</td>
<td>( \mu g/\sqrt{Hz} )</td>
</tr>
<tr>
<td>Control Timing</td>
<td>( t_{RESPONSE} )</td>
<td>—</td>
<td>1.0</td>
<td>2.0</td>
<td>ms</td>
</tr>
<tr>
<td>Power-Up Response Time(^{(6)})</td>
<td>( t_{ENABLE} )</td>
<td>—</td>
<td>0.5</td>
<td>2.0</td>
<td>ms</td>
</tr>
<tr>
<td>Enable Response Time(^{(7)})</td>
<td>( f_{CELL} )</td>
<td>—</td>
<td>6.0</td>
<td>—</td>
<td>kHz</td>
</tr>
<tr>
<td>Sensing Element Resonant Frequency</td>
<td>( f_{CELL} )</td>
<td>—</td>
<td>3.4</td>
<td>—</td>
<td>kHz</td>
</tr>
<tr>
<td>XY</td>
<td>( f_{CLK} )</td>
<td>—</td>
<td>11</td>
<td>—</td>
<td>kHz</td>
</tr>
<tr>
<td>Z</td>
<td>( f_{CLK} )</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Internal Sampling Frequency</td>
<td>( f_{CLK} )</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Output Stage Performance</td>
<td>( V_{FSO} )</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>V</td>
</tr>
<tr>
<td>Full-Scale Output Range (( I_{OUT} = 30 ) µA)</td>
<td>( V_{FSO} )</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>V</td>
</tr>
<tr>
<td>Nonlinearity, ( X_{OUT}, Y_{OUT}, Z_{OUT} )</td>
<td>( \eta_{N_{OUT}} )</td>
<td>—</td>
<td>—</td>
<td>+1.0</td>
<td>%/FSO</td>
</tr>
<tr>
<td>Cross-Axis Sensitivity(^{(8)})</td>
<td>( V_{XY, XZ, YZ} )</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>5.0</td>
</tr>
</tbody>
</table>

1. For a loaded output, the measurements are observed after an RC filter consisting of a 1.0 kΩ resistor and a 0.1 µF capacitor to ground.
2. These limits define the range of operation for which the part will meet specification.
3. Within the supply range of 2.2 and 3.6 V, the device operates as a fully calibrated linear accelerometer. Beyond these supply limits the device may operate as a linear device but is not guaranteed to be in calibration.
4. This value is measured with g-Select in 1.5g mode.
5. The device can measure both + and – acceleration. With no input acceleration the output is at mid-supply. For positive acceleration the output will increase above \( V_{DD}/2 \). For negative acceleration, the output will decrease below \( V_{DD}/2 \).
6. The response time between 10% of full scale VDD input voltage and 90% of the final operating output voltage.
7. The response time between 10% of full scale Sleep Mode input voltage and 90% of the final operating output voltage.
8. A measure of the device’s ability to reject an acceleration applied 90° from the true axis of sensitivity.
LISY300AL
MEMS inertial sensor:
single-axis ±300°/s analog output yaw rate gyroscope

Features
- 2.7 V to 3.6 V single supply operation
- Low power consumption
- Embedded power-down
- ±300 °/s full scale
- Absolute analog rate output
- Integrated low-pass filters
- Embedded self-test
- High shock survivability
- ECOPACK® RoHS and “Green” compliant (see Section 4)

Description
The LISY300AL is a low-power single-axis yaw rate sensor. It includes a sensing element and an IC interface able to provide the measured angular rate to the external world through an analog output voltage.

The sensing element, capable of detecting the yaw rate, is manufactured using a dedicated micromachining process developed by ST to produce inertial sensors and actuators on silicon wafers.

The IC interface is manufactured using a CMOS process that allows a high level of integration to design a dedicated circuit which is trimmed to better match the sensing element characteristics.

The LISY300AL has a full scale of ±300 °/s and is capable of measuring rates with a -3 dB bandwidth up to 88 Hz.

The LISY300AL is available in a plastic land grid array (LGA) package and can operate within a temperature range from -40 °C to +85 °C.

The LISY300AL belongs to a family of products suitable for a variety of applications, including:
- Gaming and virtual reality input devices
- Motion control with MMI (man-machine interface)
- Image stabilization for digital video and digital still cameras
- GPS navigation systems
- Appliances and robotics

Table 1. Device summary

<table>
<thead>
<tr>
<th>Order code</th>
<th>Temperature range (°C)</th>
<th>Package</th>
<th>Packing</th>
</tr>
</thead>
<tbody>
<tr>
<td>LISY300AL</td>
<td>-40 to +85</td>
<td>LGA-28 (7x7x1.5)</td>
<td>Tray</td>
</tr>
<tr>
<td>LISY300ALTR</td>
<td>-40 to +85</td>
<td>LGA-28 (7x7x1.5)</td>
<td>Tape and reel</td>
</tr>
</tbody>
</table>
## Contents

1. **Block diagram and pin description**  
   1.1 Pin description

2. **Mechanical and electrical specifications**  
   2.1 Mechanical characteristics
   2.2 Electrical characteristics
   2.3 Absolute maximum ratings
   2.4 Terminology  
      2.4.1 Sensitivity
      2.4.2 Zero-rate level
      2.4.3 Self-test

3. **Application hints**
   3.1 Soldering information

4. **Package information**

5. **Revision history**
1 Block diagram and pin description

The vibration of the structure is maintained by a drive circuitry in a feedback loop. The sensing signal is filtered and appears as an analog signal at the output.

1.1 Pin description

The vibration of the structure is maintained by a drive circuitry in a feedback loop. The sensing signal is filtered and appears as an analog signal at the output.
<table>
<thead>
<tr>
<th>Pin #</th>
<th>Pin Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NC</td>
<td>Internally not connected</td>
</tr>
<tr>
<td>2</td>
<td>GND</td>
<td>0V supply</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>0V supply</td>
</tr>
<tr>
<td>4</td>
<td>NC</td>
<td>Internally not connected</td>
</tr>
<tr>
<td>5</td>
<td>CACT</td>
<td>Active filter capacitor</td>
</tr>
<tr>
<td>6</td>
<td>ANALOG OUTPUT</td>
<td>Rate signal output voltage</td>
</tr>
<tr>
<td>7-9</td>
<td>NC</td>
<td>Internally not connected</td>
</tr>
<tr>
<td>10</td>
<td>PD</td>
<td>Power-down (logic 0: normal mode; logic 1: power-down mode)</td>
</tr>
<tr>
<td>11</td>
<td>ST</td>
<td>Self-test (logic 0: normal mode; logic 1: self-test)</td>
</tr>
<tr>
<td>12-13</td>
<td>Reserved</td>
<td>Leave unconnected</td>
</tr>
<tr>
<td>14-15</td>
<td>NC</td>
<td>Internally not connected</td>
</tr>
<tr>
<td>16-21</td>
<td>Reserved</td>
<td>Leave unconnected</td>
</tr>
<tr>
<td>22</td>
<td>NC</td>
<td>Internally not connected</td>
</tr>
<tr>
<td>23</td>
<td>VCONT</td>
<td>PLL filter connection pad #1</td>
</tr>
<tr>
<td>24</td>
<td>FILTVDD</td>
<td>PLL filter connection pad #2</td>
</tr>
<tr>
<td>25</td>
<td>Vdd</td>
<td>Power supply</td>
</tr>
<tr>
<td>26</td>
<td>Vdd</td>
<td>Power supply</td>
</tr>
<tr>
<td>27</td>
<td>Vdd</td>
<td>Power supply</td>
</tr>
<tr>
<td>28</td>
<td>NC</td>
<td>Internally not connected</td>
</tr>
</tbody>
</table>
2 Mechanical and electrical specifications

2.1 Mechanical characteristics

Table 3. Mechanical characteristics @ Vdd = 3.3 V, T = 25 °C unless otherwise noted(1)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Test condition</th>
<th>Min.</th>
<th>Typ.(2)</th>
<th>Max.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>FS</td>
<td>Measurement range</td>
<td>±300</td>
<td>°/s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>So</td>
<td>Sensitivity</td>
<td>3.3</td>
<td>mV/°/s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SoDr</td>
<td>Sensitivity change vs. temperature</td>
<td>From -40 °C to +85 °C</td>
<td>4</td>
<td>%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Voff</td>
<td>Zero-rate level(3)</td>
<td>1.65</td>
<td>V</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OffDr</td>
<td>Zero-rate level change vs. temperature</td>
<td>From -40 °C to +85 °C</td>
<td>5</td>
<td>°/s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NL</td>
<td>Non linearity(4)</td>
<td>±0.8</td>
<td>% FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BW</td>
<td>-3dB bandwidth(5)(5)</td>
<td>C_ACT = 10 nF</td>
<td>88</td>
<td>Hz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rn</td>
<td>Rate noise density</td>
<td>0.1</td>
<td>°/s/√Hz</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vt</td>
<td>Self-test output voltage change(6)</td>
<td>+300</td>
<td>mV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sup</td>
<td>Start-up time</td>
<td>300</td>
<td>ms</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Freq</td>
<td>Sensing element resonant frequency</td>
<td>4.5</td>
<td>kHz</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Top</td>
<td>Operating temperature range</td>
<td>-40</td>
<td>+85</td>
<td>°C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wh</td>
<td>Product weight</td>
<td>160</td>
<td>mg</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. The product is factory calibrated at 3.3 V. The operational power supply range is specified in Table 4.
2. Typical specifications are not guaranteed
3. The product is capable of sensing angular rates extending from DC to the selected bandwidth
4. Guaranteed by design
5. User selectable by external capacitor C_ACT
6. “Self-test output voltage change” is defined as Vout(Vst = logic 1) - Vout(Vst = logic 0)
## 2.2 Electrical characteristics

Table 4. Electrical characteristics @ Vdd = 3.3 V, T=25 °C unless otherwise noted(1)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Test condition</th>
<th>Min.</th>
<th>Typ.(2)</th>
<th>Max.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vdd</td>
<td>Supply voltage</td>
<td>2.7</td>
<td>3.3</td>
<td>3.6</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Idd</td>
<td>Supply current</td>
<td>PD pin connected to GND</td>
<td>4.8</td>
<td>mA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IddPdn</td>
<td>Supply current in power-down mode</td>
<td>PD pin connected to Vdd</td>
<td>1</td>
<td>µA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VST</td>
<td>Self-test input</td>
<td>Logic 0 level</td>
<td>0</td>
<td>0.2*Vdd</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>VPD</td>
<td>Power-down input</td>
<td>Logic 0 level</td>
<td>0</td>
<td>0.2*Vdd</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>CACT</td>
<td>Active low-pass filter capacitor</td>
<td>10</td>
<td>nF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVS</td>
<td>Output voltage swing(3)</td>
<td>Iout = ±100µA</td>
<td>0.4</td>
<td>Vdd-0.4</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>CLOAD</td>
<td>Capacitive load drive(3)</td>
<td>0.4</td>
<td>10</td>
<td>nF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Top</td>
<td>Operating temperature range</td>
<td>-40</td>
<td>+85</td>
<td>°C</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. The product is factory calibrated at 3.3 V
2. Typical specifications are not guaranteed
3. Referred to ANALOG OUTPUT pin #6
PMB-688 GPS module

PMB-688 FEATURES

- Built-in SiRFstarIII chipsets receivers give unparalleled GPS performance and precision. 20 parallel satellite-tracking channels for fast acquisition and reacquisition.
- Built-in WAAS/EGNOS Demodulator.
- Low power consumption and ultra mini size only 33x39mm.
- Built-in rechargeable battery for backup memory and RTC backup.
- Support NMEA0183 v2.2 data protocol.
- Enhanced algorithms providing superior navigation performance in urban, canyon and foliage environments.
- Include RF MMCX connector (Optional: Active Antenna)
**PMB-688 Specifications**

<table>
<thead>
<tr>
<th>Specification</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GPS IC</strong></td>
<td>SiRFstar III</td>
</tr>
<tr>
<td><strong>Receiver</strong></td>
<td>Tracking up to 20 satellites L1, 1575.42 MHz, C/A code</td>
</tr>
<tr>
<td><strong>Accuracy</strong></td>
<td>Position: 2DRMS approximately 5m, WAAS support</td>
</tr>
<tr>
<td></td>
<td>Velocity: 0.1 m/s without SA imposed.</td>
</tr>
<tr>
<td><strong>Time</strong></td>
<td>±1μsec</td>
</tr>
<tr>
<td><strong>Acquisition Time</strong></td>
<td>Cold Start: 42sec (Average)</td>
</tr>
<tr>
<td></td>
<td>Warm Start: 38sec (Average)</td>
</tr>
<tr>
<td></td>
<td>Hot Start: 1sec (Min.)</td>
</tr>
<tr>
<td><strong>Sensitivity</strong></td>
<td>Acquisition: -148dBm</td>
</tr>
<tr>
<td></td>
<td>Tracking: -159 dBm</td>
</tr>
<tr>
<td><strong>Dynamics</strong></td>
<td>Altitude: 18000m (Max.)</td>
</tr>
<tr>
<td></td>
<td>Velocity: 515m/s (Max.)</td>
</tr>
<tr>
<td></td>
<td>Acceleration: ±4g (Max.)</td>
</tr>
<tr>
<td><strong>Navigation update rate</strong></td>
<td>Once per second</td>
</tr>
<tr>
<td><strong>Serial Port</strong></td>
<td>TTL</td>
</tr>
<tr>
<td><strong>Baud Rate</strong></td>
<td>4800 bps (Optional 9600, 19300, 38400 bps)</td>
</tr>
<tr>
<td><strong>Output Message</strong></td>
<td>NMEA0183 V2.2 GGA, GSV, GSA, RMC (optional VTG, GLL)</td>
</tr>
<tr>
<td><strong>Datum</strong></td>
<td>WGS 84</td>
</tr>
<tr>
<td><strong>Power supply</strong></td>
<td>DC 3.3V ~ 5V</td>
</tr>
<tr>
<td><strong>Power Consumption</strong></td>
<td>Typical 65mA @12V</td>
</tr>
<tr>
<td><strong>LED Function</strong></td>
<td>Power on/off and Navigation</td>
</tr>
<tr>
<td><strong>Operating Temp.</strong></td>
<td>-20°C~+70°C</td>
</tr>
<tr>
<td><strong>Storage Temp.</strong></td>
<td>-30°C~+85°C</td>
</tr>
<tr>
<td><strong>Humidity</strong></td>
<td>5%~95%</td>
</tr>
<tr>
<td><strong>Antenna Type</strong></td>
<td>Built-in patch antenna</td>
</tr>
<tr>
<td><strong>RF Connector</strong></td>
<td>MMCX type (Optional: Active Antenna)</td>
</tr>
</tbody>
</table>
1. Overview

This manual describes the operation of the XBee/XBee-PRO ZB RF module, which consists of ZigBee firmware loaded onto XBee S2 and S2B hardware, models: XBEE2, XBEEPRO2 and PRO S2B. The XBee/XBee-PRO ZB RF Modules are designed to operate within the ZigBee protocol and support the unique needs of low-cost, low-power wireless sensor networks. The modules require minimal power and provide reliable delivery of data between remote devices.

The modules operate within the ISM 2.4 GHz frequency band and are compatible with the following:

- XBee RS-232 Adapter
- XBee RS-485 Adapter
- XBee Analog I/O Adapter
- XBee Digital I/O Adapter
- XBee Sensor
- XBee USB Adapter
- XStick
- ConnectPort X Gateways
- XBee Wall Router.

The XBee/XBee-PRO ZB firmware release can be installed on XBee ZNet or ZB modules. The XBee ZB firmware is based on the EmberZNet 3.x ZigBee PRO Feature Set mesh networking stack, while the XBee ZNet 2.5 firmware is based on Ember's proprietary "designed for ZigBee" mesh stack (EmberZNet 2.5.x). ZB and ZNet 2.5 firmware are similar in nature, but not over-the-air compatible. Devices running ZNet 2.5 firmware cannot talk to devices running the ZB firmware.

What's New in 2x7x

Firmware

XBee/XBee-PRO ZB firmware includes the following features (compared with 2x6x):

- Using Ember stack version 3.4.1.
- Support for the PRO S2B with temperature compensation and an overvoltage check. Within 15 seconds of the supply voltage exceeding 3.9V, the API will emit a 0x08 modem status (Overvoltage) message, and then the AT/API versions will do a watchdog reset.
- ZDO pass-through added. If AO=3, then ZDO requests which are not supported by the stack will be passed out the UART.
- An attempt to send an oversized packet (256+ bytes) will result in a Tx Status message with a status code of 0x74.
- End devices have two speed polling. 7.5 seconds is the slow rate, which switches to the fast rate to transact with its parent. When transactions are done, it switches back to the slow rate.
- A new receive option bit (0x40) indicates if the packet came from an end device.
- Added extended timeout option since end devices need more time than routers to ack their packets.
- An option bit (0x01) was added to disable APS retries.
- If an end device has not had its polls answered for 5 secs, it will leave and attempt to rejoin the network.
- XBee S2B has a new TP command which returns the temperature compensation sensor reading in units of Celsius degrees.
- The PP command returns the power dBm setting when PL4 is selected.
- The PO command sets the slow polling rate on end devices. Range is 1-0x1770 in units of 10 msec (10 msec to 60 sec). Default is 0 which invokes a 100 msec delay.
- Rejoining now can proceed without a NR or NRO command after a Mgmt_Leave_req is processed.
- Command ranges were changed for the SC, IR, and LT commands.
- A PAN ID corruption problem was fixed.

See the 2x7x release notes for a complete list of new features and bug fixes at www.digi.com/support.
The XBee/XBee-PRO/S2B ZB 2x7x manual includes the following corrections over the 2x6x manual:

- Descriptions and specification for the PRO S2B.
- SIF Header Interface, pin 8 relabeled as pin 10.
- Pin mappings for pins 22 and 24 updated.
- New modem status codes were added.
- Corrections to the ZigBee Receive Packet description.
- Description changes for the SC, PL, PP, AO, IR, %V, and PO commands.
- Updates to Appendix B.
## Key Features

<table>
<thead>
<tr>
<th>High Performance, Low Cost</th>
<th>Low Power</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>XBee</strong></td>
<td><strong>XBee</strong></td>
</tr>
<tr>
<td>• Indoor/Urban: up to 133' (40 m)</td>
<td>• TX Peak Current: 40 mA (@3.3 V)</td>
</tr>
<tr>
<td>• Outdoor line-of-sight: up to 400' (120 m)</td>
<td>• RX Current: 40 mA (@3.3 V)</td>
</tr>
<tr>
<td>• Transmit Power: 2 mw (3 dBm)</td>
<td>• Power-down Current: &lt; 1 μA</td>
</tr>
<tr>
<td>• Receiver Sensitivity: -96 dBm</td>
<td><strong>XBee-PRO (S2)</strong></td>
</tr>
<tr>
<td></td>
<td>• TX Peak Current: 295mA (170mA for International variant)</td>
</tr>
<tr>
<td><strong>XBee-PRO (S2)</strong></td>
<td>• RX Current: 45 mA (@3.3 V)</td>
</tr>
<tr>
<td>• Indoor/Urban: up to 300' (90 m), 200' (60 m) for International variant</td>
<td>• Power-down Current: 3.5 μA typical @ 25 degrees C</td>
</tr>
<tr>
<td>• Outdoor line-of-sight: up to 2 miles (3200 m), 5000' (1500 m) for International variant</td>
<td><strong>XBee-PRO (S2B)</strong></td>
</tr>
<tr>
<td>• Transmit Power: 50mW (17dBm), 10mW (10dBm) for International variant</td>
<td>• TX Peak Current: 205mA (117mA for International variant)</td>
</tr>
<tr>
<td>• Receiver Sensitivity: -102 dBm</td>
<td>• RX Current: 47 mA (@3.3 V)</td>
</tr>
<tr>
<td><strong>XBee-PRO (S2B)</strong></td>
<td>• Power-down Current: 3.5 μA typical @ 25 degrees C</td>
</tr>
<tr>
<td>• Indoor/Urban: up to 300' (90 m), 200' (60 m) for International variant</td>
<td><strong>Easy-to-Use</strong></td>
</tr>
<tr>
<td>• Outdoor line-of-sight: up to 2 miles (3200 m), 5000' (1500 m) for International variant</td>
<td>No configuration necessary for out-of-box RF communications</td>
</tr>
<tr>
<td>• Transmit Power: 63mW (18dBm), 10mW (10dBm) for International variant</td>
<td>AT and API Command Modes for configuring module parameters</td>
</tr>
<tr>
<td>• Receiver Sensitivity: -102 dBm</td>
<td>Small form factor</td>
</tr>
<tr>
<td><strong>Advanced Networking &amp; Security</strong></td>
<td>Extensive command set</td>
</tr>
<tr>
<td>Retries and Acknowledgements</td>
<td>Free X-CTU Software (Testing and configuration software)</td>
</tr>
<tr>
<td>DSSS (Direct Sequence Spread Spectrum)</td>
<td>Free &amp; Unlimited Technical Support</td>
</tr>
<tr>
<td>Each direct sequence channel has over 65,000 unique network addresses available</td>
<td></td>
</tr>
<tr>
<td>Point-to-point, point-to-multipoint and peer-to-peer topologies supported</td>
<td></td>
</tr>
<tr>
<td>Self-routing, self-healing and fault-tolerant mesh networking</td>
<td></td>
</tr>
</tbody>
</table>

### Worldwide Acceptance

FCC Approval (USA) Refer to Appendix A for FCC Requirements. Systems that contain XBee®/XBee-PRO® ZB RF Modules inherit Digi Certifications.

ISM (Industrial, Scientific & Medical) 2.4 GHz frequency band

Manufactured under ISO 9001:2000 registered standards

XBee®/XBee-PRO® ZB RF Modules are optimized for use in US, Canada, Europe, Australia, and Japan (contact Digi for complete list of agency approvals).
## Specifications of the XBee®/XBee-PRO® ZB RF Module

<table>
<thead>
<tr>
<th>Specification</th>
<th>XBee (S2)</th>
<th>XBee-PRO (S2)</th>
<th>XBee-PRO (S2B)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performance</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indoor/Urban Range</td>
<td>Up to 133 ft. (40 m)</td>
<td>Up to 300 ft. (90 m), up to 200 ft (60 m)</td>
<td>Up to 300 ft. (90 m), up to 200 ft (60 m)</td>
</tr>
<tr>
<td>Outdoor RF line-of-sight Range</td>
<td>Up to 400 ft. (120 m)</td>
<td>Up to 2 miles (3200 m), up to 5000 ft (1500 m)</td>
<td>Up to 2 miles (3200 m), up to 5000 ft (1500 m)</td>
</tr>
<tr>
<td>Transmit Power Output</td>
<td>2mW (+3dBm), boost mode enabled</td>
<td>1.25mW (+1dBm), boost mode disabled</td>
<td>10mW (+10 dBm) for International variant</td>
</tr>
<tr>
<td>RF Data Rate</td>
<td>250,000 bps</td>
<td>250,000 bps</td>
<td>250,000 bps</td>
</tr>
<tr>
<td>Data Throughput</td>
<td>up to 35000 bps (see chapter 4)</td>
<td>up to 35000 bps (see chapter 4)</td>
<td>up to 35000 bps (see chapter 4)</td>
</tr>
<tr>
<td><strong>Serial Interface Data Rate (software selectable)</strong></td>
<td>1200 bps - 1 Mbps (non-standard baud rates also supported)</td>
<td>1200 bps - 1 Mbps (non-standard baud rates also supported)</td>
<td>1200 bps - 1 Mbps (non-standard baud rates also supported)</td>
</tr>
<tr>
<td><strong>Receiver Sensitivity</strong></td>
<td>-96 dBm, boost mode enabled</td>
<td>-102 dBm</td>
<td>-102 dBm</td>
</tr>
<tr>
<td><strong>Power Requirements</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supply Voltage</td>
<td>2.1 - 3.6 V</td>
<td>3.0 - 3.4 V</td>
<td>2.7 - 3.6 V</td>
</tr>
<tr>
<td>Operating Current (Transmit, max output power)</td>
<td>40mA (@3.3 V, boost mode enabled)</td>
<td>295mA (@3.3 V) 170mA (@3.3 V)</td>
<td>International variant</td>
</tr>
<tr>
<td>Operating Current (Receive)</td>
<td>40mA (@3.3 V, boost mode enabled)</td>
<td>45 mA (@3.3 V)</td>
<td>47 mA, up to 82 mA with programmable variant (@3.3 V)</td>
</tr>
<tr>
<td>Idle Current (Receiver off)</td>
<td>15mA</td>
<td>15mA</td>
<td>15mA</td>
</tr>
<tr>
<td>Power-down Current</td>
<td>&lt; 1 μA @ 25°C</td>
<td>3.5 μA typical @ 25°C</td>
<td>3.5 μA typical @ 25°C</td>
</tr>
<tr>
<td><strong>General</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating Frequency Band</td>
<td>ISM 2.4 GHz</td>
<td>ISM 2.4 GHz</td>
<td>ISM 2.4 GHz</td>
</tr>
<tr>
<td>Dimensions</td>
<td>0.960&quot; x 1.087&quot; (2.438cm x 2.761cm)</td>
<td>0.960 x 1.297 (2.438cm x 3.294cm)</td>
<td>0.960 x 1.297 (2.438cm x 3.294cm)</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>-40 to 85°C (industrial)</td>
<td>-40 to 85°C (industrial)</td>
<td>-40 to 85°C (industrial)</td>
</tr>
<tr>
<td>Antenna Options</td>
<td>Integrated Whip, Chip, RPSMA, or U.FL Connector</td>
<td>Integrated Whip, Chip, RPSMA, or U.FL Connector</td>
<td>Integrated Whip,PCB Embedded Trace, RPSMA, or U.FL Connector</td>
</tr>
<tr>
<td><strong>Networking &amp; Security</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supported Network Topologies</td>
<td>Point-to-point, Point-to-multipoint, Peer-to-peer, and Mesh</td>
<td>Point-to-point, Point-to-multipoint, Peer-to-peer, and Mesh</td>
<td>Point-to-point, Point-to-multipoint, Peer-to-peer, and Mesh</td>
</tr>
<tr>
<td>Number of Channels</td>
<td>16 Direct Sequence Channels</td>
<td>14 Direct Sequence Channels</td>
<td>15 Direct Sequence Channels</td>
</tr>
<tr>
<td>Channels</td>
<td>11 to 26</td>
<td>11 to 24</td>
<td>11 to 25</td>
</tr>
<tr>
<td>Addressing Options</td>
<td>PAN ID and Addresses, Cluster IDs and Endpoints (optional)</td>
<td>PAN ID and Addresses, Cluster IDs and Endpoints (optional)</td>
<td>PAN ID and Addresses, Cluster IDs and Endpoints (optional)</td>
</tr>
<tr>
<td><strong>Agency Approvals</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Industry Canada (IC)</td>
<td>IC: 4214A-XBEE2</td>
<td>IC: 1846A-XBEEPRO2</td>
<td>IC: 1846A-PROS2B</td>
</tr>
<tr>
<td>Europe (CE)</td>
<td>ETSI</td>
<td>ETSI (international variant)</td>
<td>ETSI (10 mW max)</td>
</tr>
</tbody>
</table>

© 2010 Digi International, Inc.
Hardware Specs for Programmable Variant

The following specifications need to be added to the current measurement of the previous table if the module has the programmable secondary processor. For example, if the secondary processor is running and constantly collecting DIO samples at a rate while having the RF portion of the XBEE sleeping the new current will be \( I_{\text{total}} = I_{\text{r2}} + I_{s} \). Where \( I_{r2} \) is the runtime current of the secondary processor and \( I_{s} \) is the sleep current of the RF portion of the module of the XBEE-PRO (S2B) listed in the table below.

Specifications of the programmable secondary processor

<table>
<thead>
<tr>
<th>Optional Secondary Processor Specification</th>
<th>These numbers add to S2B specifications (Add to RX, TX, and sleep currents depending on mode of operation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Runtime current for 32k running at 20MHz</td>
<td>+14mA</td>
</tr>
<tr>
<td>Runtime current for 32k running at 1MHz</td>
<td>+1mA</td>
</tr>
<tr>
<td>Sleep current</td>
<td>+0.5\mu A typical</td>
</tr>
<tr>
<td>For additional specifications see Freescale Datasheet and Manual</td>
<td>1.8VDC to VCC</td>
</tr>
<tr>
<td>Minimum Reset low pulse time for EM250</td>
<td>+50 nS (additional resistor increases minimum time)</td>
</tr>
</tbody>
</table>

Mechanical Drawings

Mechanical drawings of the XBee®/XBee-PRO® ZB RF Modules (antenna options not shown)
The XBee/XBee-PRO ZB modules include a SIF programming header that can be used with Ember’s programming tools to upload custom firmware images onto the XBee module. The SIF header orientation and pinout are shown below.

A male header can be populated on the XBee that mates with Ember’s 2x5 ribbon cable. The male header and ribbon cables are available from Samtec:

- 2x5 Male Header - FTSH-105-01-F-DV-K
- 2x5 Ribbon Cable - FFSD-05-D-12.00-01-N
Mounting Considerations

The XBee modules were designed to mount into a receptacle (socket) and therefore does not require any soldering when mounting it to a board. The XBee-PRO Development Kits contain RS-232 and USB interface boards which use two 20-pin receptacles to receive modules.

XBeepro Module Mounting to an RS-232 Interface Board.

The receptacles used on Digi development boards are manufactured by Century Interconnect. Several other manufacturers provide comparable mounting solutions; however, Digi currently uses the following receptacles:

- Through-hole single-row receptacles - Samtec P/N: MMS-110-01-L-SV (or equivalent)
- Surface-mount double-row receptacles - Century Interconnect P/N: CPRMSL20-D-0-1 (or equivalent)
- Surface-mount single-row receptacles - Samtec P/N: SMM-110-02-SM-S

Digi also recommends printing an outline of the module on the board to indicate the orientation the module should be mounted.
## Pin Signals

### Pin Assignments for the XBee-PRO Modules

(Low-asserted signals are distinguished with a horizontal line above signal name.)

<table>
<thead>
<tr>
<th>Pin #</th>
<th>Name</th>
<th>Direction</th>
<th>Default State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VCC</td>
<td>-</td>
<td>-</td>
<td>Power supply</td>
</tr>
<tr>
<td>2</td>
<td>DOUT</td>
<td>Output</td>
<td>Output</td>
<td>UART Data Out</td>
</tr>
<tr>
<td>3</td>
<td>DIN / CONFIG</td>
<td>Input</td>
<td>Input</td>
<td>UART Data In</td>
</tr>
<tr>
<td>4</td>
<td>DIO12</td>
<td>Both</td>
<td>Disabled</td>
<td>Digital I/O 12</td>
</tr>
<tr>
<td>5</td>
<td>RESET</td>
<td>Both</td>
<td>Open-Collector with pull-up</td>
<td>Module Reset (reset pulse must be at least 200 ns)</td>
</tr>
<tr>
<td>6</td>
<td>RSSI PWM / DIO10</td>
<td>Both</td>
<td>Output</td>
<td>RX Signal Strength Indicator / Digital I/O</td>
</tr>
<tr>
<td>7</td>
<td>DIO11</td>
<td>Both</td>
<td>Input</td>
<td>Digital I/O 11</td>
</tr>
<tr>
<td>8</td>
<td>[reserved]</td>
<td>-</td>
<td>Disabled</td>
<td>Do not connect</td>
</tr>
<tr>
<td>9</td>
<td>DTR / SLEEP_RQ / DIO8</td>
<td>Both</td>
<td>Input</td>
<td>Pin Sleep Control Line or Digital I/O 8</td>
</tr>
<tr>
<td>10</td>
<td>GND</td>
<td>-</td>
<td>-</td>
<td>Ground</td>
</tr>
<tr>
<td>11</td>
<td>DIO4</td>
<td>Both</td>
<td>Disabled</td>
<td>Digital I/O 4</td>
</tr>
<tr>
<td>12</td>
<td>CTS / DIO7</td>
<td>Both</td>
<td>Output</td>
<td>Clear-to-Send Flow Control or Digital I/O 7. CTS, if enabled, is an output.</td>
</tr>
<tr>
<td>13</td>
<td>ON / SLEEP</td>
<td>Output</td>
<td>Output</td>
<td>Module Status Indicator or Digital I/O 9</td>
</tr>
<tr>
<td>14</td>
<td>VREF</td>
<td>Input</td>
<td>-</td>
<td>Not used for EM250. Used for programmable secondary processor. For compatibility with other XBEE modules, we recommend connecting this pin voltage reference if Analog sampling is desired. Otherwise, connect to GND.</td>
</tr>
<tr>
<td>15</td>
<td>Associate / DIO5</td>
<td>Both</td>
<td>Output</td>
<td>Associated Indicator, Digital I/O 5</td>
</tr>
<tr>
<td>16</td>
<td>RTS / DIO6</td>
<td>Both</td>
<td>Input</td>
<td>Request-to-Send Flow Control, Digital I/O 6. RTS, if enabled, is an input.</td>
</tr>
<tr>
<td>17</td>
<td>AD3 / DIO3</td>
<td>Both</td>
<td>Disabled</td>
<td>Analog Input 3 or Digital I/O 3</td>
</tr>
<tr>
<td>18</td>
<td>AD2 / DIO2</td>
<td>Both</td>
<td>Disabled</td>
<td>Analog Input 2 or Digital I/O 2</td>
</tr>
<tr>
<td>19</td>
<td>AD1 / DIO1</td>
<td>Both</td>
<td>Disabled</td>
<td>Analog Input 1 or Digital I/O 1</td>
</tr>
<tr>
<td>20</td>
<td>AD0 / DIO0 / Commissioning Button</td>
<td>Both</td>
<td>Disabled</td>
<td>Analog Input 0, Digital I/O 0, or Commissioning Button</td>
</tr>
</tbody>
</table>

- Signal Direction is specified with respect to the module
- See Design Notes section below for details on pin connections.
OpenLog Datasheet

OpenLog is a simple serial logger based on the ATmega328 running at 16MHz. The ATmega328 should be able to talk to high capacity (larger than 2GB) SD cards. The whole purpose of this logger was to create a logger that just powered up and worked. OpenLog ships with a standard serial bootloader so you can load new firmware with a simple serial connection.

The Basics

- OpenLog runs at 3.3-5V at 9600bps by default. This is configurable to 2400, 4800, 9600, 19200, 57600, and 115200bps. We recommend you attach a serial connection to reconfigure the unit to work at a different serial speed, but you should be able to do it in software.
- The microSD card can be any size from 64MB to 16GB. Before using OpenLog be sure to format the card either FAT16 or FAT32. We recommend using windows to format your card. If using Linux, be sure to create a DOS filesystem after formatting the card.
- During power up, you will see 12> or 12<. 1 indicates the serial connection is established. 2 indicates the SD card has been successfully initialized.
  - ‘<’ indicates OpenLog is ready and will log any serial data received (this is the default mode)
  - ‘>’ indicates OpenLog is ready to receive commands.
- Type ? at the > prompt to bring up a list of supported commands.
- If you are actively logging in NewLog or SeqLog mode, sending Ctrl+z (ASCII 26) three times will exit logging mode and enter command mode.
- For a full list of commands, see the Command Set page.

Connections

- **GRN**: Reset pin and connects to the GRN pin on the Arduino Pro Mini. Pulling this line low will reset the ATmega328. Because there is a capacitor on this line, holding this line low will not keep OpenLog in reset.
- **RXI**: Serial input into OpenLog.
- **TXO**: Serial output from OpenLog.
- **VCC**: 3.3V to 12V input. We recommend 3.3V to 5V.
- **GND**: Ground
- **BLK**: This pin is connected to GND. Connect this pin to BLK on the Arduino Pro Mini.

The four pins shown at the top of the board are connected to the SPI pins for programming. We use a special pogo-pin jig to program the serial bootloader onto each board. You are welcome to connect to them but realize the SPI pins are shared with the interface to the SD socket so you might not want to use them as GPIOs.
Status LEDs

- STAT1 LED is the LED shown above, right of the word OpenLog and is sitting on PD5 (Arduino D5)
  This LED toggles on/off every time a new character is received. This LED helps troubleshoot and indicate serial communication is working.
- STAT2 LED is the LED shown above, left of the word OpenLog and is sitting on PB5 (Arduino D13)
  This LED is attached to the SPI Serial Clock line. You will see this LED flash rarely. It only turns on when the SPI interface is active and this is rare as the OpenLog buffers 512 bytes at a time before recording to the SD card. Recording 512 bytes is very fast so the LED is on for very little.

Features

- Supports automatic log generation and recording. Turn on OpenLog, wait ~2 seconds and start throwing text at it!
- Supports 2400, 4800, 9600, 19200, 57600, and 115200 serial baud rates at 8-N-1.
- Supports creating a new file or directory up to 16 characters including the ‘.’ and extension. “123456789012_.txt” is the longest name. Any longer and the module will re-initialize as if there was a variable sizeof error.
- Capital letters, white space, and other characters are supported (“Hi there#$_.txt”).
- Recording constant 115200bps datastreams are supported. Throw it everything you’ve got!
- The change directory command is a bit weird. Normally it’s ‘cd.’ but to change to a lower dir, use ‘cd ..’ (space between cd and ..)
- If you get OpenLog stuck into an unknown baudrate, there is a safety mechanism built-in. Tie the RX pin to ground and power up OpenLog. You should see the LEDs blink back and forth for 2 seconds, then blink in unison. Now power down OpenLog and remove the RX/GND jumper. OpenLog is now reset to 9600bps with an escape character of ctrl+z sent three consecutive times.
- Pre-programmed STK500 (Arduino compatible) serial bootloader running at 57600bps @ 16MHz
  Please note: The preloaded STK500 serial bootloader is 2k bytes of flash, and begins at 0x7800 (30,720). If the code is larger than 30,719 bytes, you will get verification errors during serial bootloading. **Warning:** some early units (sold in December of 2009) of OpenLog did not have a bootblock protection lock bit set and will overwrite the bootloader. To check if you have one of these versions, drop to command mode and type ‘?’ The OpenLog firmware version number is shown at the top of the menu. Everything after v1.0 (v1.1 and above) is good and does not have this problem. SD vs SDHC configuration is found in sd_raw_config.h – Currently only 512MB, 1GB, 2GB, and some 4GB cards work (not yet compatible with SDHC cards). Merely enabling SDHC in the current versions will cause the code to get very large so it won’t fit. (A parallel project for FAT32 support includes SDHC, fits in the available space and is at http://github.com/tz1/sparkfun – hyperlog and fat32cli are the openlog variants but are not directly compatible).
Power
Input voltage on VCC can be 3.3 to 12V. Input voltage on RXI pin must not exceed 6V. Output voltage on TXO pin will not be greater than 3.3V. This may cause problems with some systems – for example if your attached microcontroller requires 4V minimum for serial communication (this is rare).

OpenLog has reverse power protection. The Micrel voltage regulator can take some serious abuse (reverse power applied, over current shut down, over voltage protection). All parts are static sensitive, but the ATmega328 and Micrel regulator have built-in static protection.

Current consumption:
- 2mA Idle
- 6mA Actively writing to a file

6mA is rare. The vast majority of the time OpenLog is idle. Writing to the SD card (6mA) happens once a 512 byte buffer fills up. Recording that buffer completes in a fraction of a second so the average consumption is very near 5mA unless you are pounding the serial port at 115200bps with a constant data stream.

Note: OpenLog may lose characters if power is removed. During an append, OpenLog will buffer 512 characters at a time. That means that if the system loses power while reading in characters, you may loose up to, but no more than, 511 characters. This is important for low power systems where you may not know when the battery or power will die. OpenLog should record each buffer as it receives each 512 byte chunk. The only way to exit an append is with Ctrl+z (ASCII 26). In firmware v1.3 and above, OpenLog has an auto-save feature. If OpenLog is idle for 5 seconds or more, it will auto-save any characters in the buffer. This is very helpful for systems that store a few characters every few seconds. This feature also significantly saves on power.

Troubleshooting
The easiest way to get OpenLog working is with a serial connection to a computer. Power up OpenLog and you should see "12<". If you don't, make sure your TXO and RXI pins are connected correctly. TXO is an output pin from OpenLog and will need to be connected to an input pin on your serial conversion board.

I don't know what baud rate I put it into! Help! : Emergency reset – aka factory defaults. If you get OpenLog stuck into an unknown baud rate, there is a safety mechanism built-in. Tie the RX pin to ground and power up OpenLog. You should see the LEDs blink back and forth for 2 seconds, then blink in unison. Now power down OpenLog and remove the RX/GND jumper. OpenLog is now reset to 9600bps. After a power up you should see ‘12<’. To get OpenLog into command mode, press ctrl+z three times.

OpenLog communicates with TTL, not RS232, because it is meant to be connected with a microcontroller or an embedded project. If you are connecting OpenLog to a computer, you will need
a TTL-to-RS232 converter board such as the RS232 Shifter board, FTDI Basic, or the FT2332 Breakout.

OpenLog has two onboard LEDs. STAT1 will blink with an error code if something is wrong. Currently there are two error codes:

- 3 Blinks: The SD card failed to initialize. You may need to format the card with FAT/FAT16 on a computer.
- 5 Blinks: OpenLog has changed to a new baud rate and needs to be power cycled.

The Card Detect feature of the microSD socket is connected to the ATmega328 but we are not currently checking for physical presence. This is because there are some SD sockets that have this feature available, and some that do not. When we began production of OpenLog we were not sure which socket would be available so we skipped this check in the firmware.

**What is the limit on the number of files I can create in the root directory?**
Currently the log number limit is 65,534. You can load hundreds of files into the root directory, but OpenLog will perform more and more slowly as more files are introduced. We recommend logging to a freshly formatted FAT16 microSD card. 10 or 20 files/logs is fine. When you approach 300, OpenLog can take multiple seconds to create a new file and start to log.

**What is the limit of sub-directories I can create?**
Unknown at this time

**Example Arduino Code:**
The easiest way to use OpenLog with an Arduino is to simply attach the RXI pin on the OpenLog to the TX pin on the Arduino. Anything that the Arduino outputs (sensor readings, GPS coordinates, etc) will be recorded.

```cpp
Serial.begin(9600); //9600bps is default for OpenLog
Serial.println("123");
```

will cause the Arduino to output ‘123’ at 9600bps and will be logged by OpenLog. An example Arduino sketch is available [here](#).

**Example C Code:**
You will need to setup your microcontroller to output serial streams at 9600bps 8N1. Almost all microcontrollers now have a UART and are configurable for this setup.

**How do I attach a FTDI Basic board to OpenLog for configuring and bootloading?**
You can not attach an FTDI Basic or FTDI Cable directly. This is because you have to swap TX and RX.

Note that these images are not to scale. The FTDI board is exactly the width of OpenLog. If you can generate a nicer image please do so!
Thanks for the live shot Michael! Notice that he uses unsoldered wires. This works but can be a bit finicky (disclaimer).

A picture showing OpenLog in a bread board or a clean diagram using Fritzing would rock. If you can create some of these images, it would help out a lot. Please help!

But once you’ve swapped TX/RX, you can easily use an FTDI Basic to talk to, configure, and quickly bootload new firmware onto OpenLog.

**Why in the world did you do mess up TX and RX like that?**

Most of the data logging projects (such as logging the temperature of your compost pile over 3 months) take place away from a computer. Therefore, OpenLog will probably not be connected to a computer – instead, it will likely be connected to a microcontroller. We made OpenLog so that it can plug directly onto an Arduino Mini Pro

**How do I upgrade the firmware on OpenLog?**

See the [Flashing Firmware](#) page for more information.

**How do I compile OpenLog?**

Before you get started, ask yourself, do I want version 1 or version 2 of OpenLog firmware?

**Version 1:**
- Is stable
- Works with cards up to 2GB
- Works only with FAT16
- Doesn’t drop characters at 57600
- Requires these instructions and steps to update the firmware

**Version 2:**
- Is currently stable, but we’re always adding things that may break it.
- Works with cards up to 16GB
- Works automagically with SD, SDHC, FAT16 and FAT32
- May drop characters at 115200bps and 57600bps
- Easy to update firmware through Arduino, loading as a sketch, but requires special changes to Arduino installation
- The Arduino IDE makes it much easier for folks to make changes

**Compile Version 2**

The following instructions are for loading v2.4 of the firmware (the current version) onto an OpenLog.

For OpenLog to work under Arduino, we have to remove the normal serial interrupt that Serial.available() uses and replace it with our own RX interrupt vector in the main sketch.

To compile OpenLog you will need to install a fresh copy of Arduino somewhere. I recommend `C:\arduino-OpenLog\` so that you don’t confuse the installs. Once you’ve unzipped all the Arduino files to `C:\arduino-OpenLog\`, replace the file located here:
with the HardwareSerial.cpp that you got when you did the github pull.

Now open Arduino.exe from this new C:\arduino-OpenLog directory. Open the OpenLog_v2 sketch from the Arduino IDE. Make any changes you’d like. Now simply compile and upload under the Arduino IDE.

Remember, compiling normal Sketches (non-OpenLog firmware) using this special version of Arduino will drive you mad because the serial interrupts will not work. I use two different icons on my desktop to differentiate between the two (Arduino 0021 and Arduino OpenLog).

Remember, you will need to swap TX and RX to upload from a normal FTDI Basic board.

Compile Version 1
The following instructions are for loading v1.61 of the firmware (the older version) onto an OpenLog:

OpenLog version 1 is compiled with WinAVR but should be compatible with any platform that support the ATmega328. I will assume you are using WinAVR.

Download all the files from the main page. Open a command prompt, navigate to the directory that contains OpenLog and type make all. You should see GCC complete without errors and create main.hex. This is the file that you will bootload using avrdude and a serial connection. See the Serial bootloading an Arduino board section of this tutorial for more information on bootloading.
CMPS09 - Tilt Compensated Compass Module

Serial mode

Connections

To use the serial mode of operation the mode pin must be connected to ground.

Commands

Below is a table describing commands that can be sent to the CMPS09 and the data it will respond with.

<table>
<thead>
<tr>
<th>Command</th>
<th>Name</th>
<th>Bytes returned</th>
<th>Returned data description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x11</td>
<td>GET VERSION</td>
<td>1</td>
<td>Software version</td>
</tr>
<tr>
<td>0x12</td>
<td>GET ANGLE 8 BIT</td>
<td>1</td>
<td>Angle as a single byte 0-255</td>
</tr>
<tr>
<td>0x13</td>
<td>GET ANGLE 16 BIT</td>
<td>2</td>
<td>Angle as two bytes, high byte first 0-3600</td>
</tr>
<tr>
<td>0x14</td>
<td>GET PITCH</td>
<td>1</td>
<td>Pitch angle +/- 0-85°</td>
</tr>
<tr>
<td>0x15</td>
<td>GET ROLL</td>
<td>1</td>
<td>Roll angle +/- 0-85°</td>
</tr>
<tr>
<td>0x21</td>
<td>GET MAG RAW</td>
<td>6</td>
<td>Raw magnetic data, 16 bit signed: X high, X low, Y high, Y low, Z high, Z low</td>
</tr>
<tr>
<td>0x22</td>
<td>GET ACCEL RAW</td>
<td>6</td>
<td>Raw accelerometer data, 16 bit signed: X high, X low, Y high, Y low, Z high, Z low</td>
</tr>
<tr>
<td>0x23</td>
<td>GET ALL</td>
<td>4</td>
<td>angle high, angle low (0-3600), pitch +/- 0-85, roll +/- 0-85</td>
</tr>
<tr>
<td>0x31</td>
<td>CALIBRATE EN1</td>
<td>1</td>
<td>returns ok (0x55)</td>
</tr>
<tr>
<td>0x45</td>
<td>CALIBRATE EN2</td>
<td>1</td>
<td>returns ok (0x55)</td>
</tr>
<tr>
<td>0x5A</td>
<td>CALIBRATE EN3</td>
<td>1</td>
<td>returns ok (0x55)</td>
</tr>
<tr>
<td>0x5E</td>
<td>CALIBRATE</td>
<td>1</td>
<td>returns ok (0x55)</td>
</tr>
<tr>
<td>0x6A</td>
<td>RESTORE 1</td>
<td>1</td>
<td>returns ok (0x55)</td>
</tr>
<tr>
<td>0x7C</td>
<td>RESTORE 2</td>
<td>1</td>
<td>returns ok (0x55)</td>
</tr>
<tr>
<td>0x81</td>
<td>RESTORE 3</td>
<td>1</td>
<td>returns ok (0x55)</td>
</tr>
<tr>
<td>0xA0</td>
<td>BAUD 19200</td>
<td>1</td>
<td>returns ok (0x55)</td>
</tr>
<tr>
<td>0xA1</td>
<td>BAUD 38400</td>
<td>1</td>
<td>returns ok (0x55)</td>
</tr>
</tbody>
</table>
**Communication settings**
The Serial mode operates over a link with a default baud rate of 9600 bps (no parity, 2 stop bits) and 3.3v-5v signal levels.
This is not RS232. Do not connect RS232 to the module, the high RS232 voltages will irreversibly damage the module.

**Calibration the CMPS09**
I would recommend evaluating the CMPS09 performance first before implementing this function. Its purpose is to remove offsets caused by constant magnetic sources around the CMPS09. First of all you need to determine North and align the CMPS09 with it, then write a sequence of 3 commands in the correct order with a small delay between bytes, 100ms will be more than adequate. The sequence to enter calibration mode is 0x31,0x45,0x5A, then calibrate the first point by sending 0x5E to the command register, this should also light the LED. The Compass should then be rotated 90° and 0x5E sent to the command register again, repeat for two further 90° rotations and the calibration completes and the LED turns off. Please make sure that the CMPS09 is not located near to ferrous objects as this will distort the magnetic field and induce errors in the reading.

**Restore of factory calibration of the CMPS09**
To perform a restore of the factory calibration write a sequence of 3 commands in the correct order with a small delay between bytes, 100ms will be more than adequate. The sequence is 0x6A,0x7C,0x81.

**Changing the baud rate**
While the CMPS09 operates at a default serial bus baud rate of 9600 you may wish to change this. There are two other baud rates that can be used, for 19200 just send 0xA0 or alternatively for 38400 send 0xA1. Please note that the CMPS09 will always default to its 9600kbps rate after power cycling and after setting a new baud rate the ok response (0x55) will be sent at the newly selected speed.
Integrated Silicon Pressure Sensor
On-Chip Signal Conditioned,
Temperature Compensated and
Calibrated

The MPX5999D piezoresistive transducer is a state-of-the-art pressure
sensor designed for a wide range of applications, but particularly for those
employing a microcontroller or microprocessor with A/D inputs. This
patented, single element transducer combines advanced micromachining
techniques, thin-film metallization and bipolar semiconductor processing to
provide an accurate, high level analog output signal that is proportional to
applied pressure.

Features

• Temperature Compensated Over 0 to 85°C
• Ideally Suited for Microprocessor or Microcontroller-Based Systems
• Patented Silicon Shear Stress Strain Gauge
• Durable Epoxy Unibody Element

<table>
<thead>
<tr>
<th>Device Name</th>
<th>Case No.</th>
<th># of Ports</th>
<th>Pressure Type</th>
<th>Device Marking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unibody Package</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPX5999D</td>
<td>867</td>
<td>-</td>
<td>Gauge</td>
<td>Absolute</td>
</tr>
</tbody>
</table>

UNIBODY PACKAGE

MPX5999D
CASE 867
## Pressure Operating Characteristics

Table 1. Operating Characteristics \((V_S = 5.0 \text{ Vdc}, T_A = 25^\circ \text{C} \text{ unless otherwise noted, } P_1 > P_2. \text{ Decoupling circuit shown in Figure 4 required to meet electrical specifications.})\)

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Symbol</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pressure Range(^{(1)})</td>
<td>(P_{OP})</td>
<td>0</td>
<td>—</td>
<td>1000</td>
<td>kPa</td>
</tr>
<tr>
<td>Supply Voltage(^{(2)})</td>
<td>(V_S)</td>
<td>4.75</td>
<td>5.0</td>
<td>5.25</td>
<td>Vdc</td>
</tr>
<tr>
<td>Supply Current</td>
<td>(I_C)</td>
<td>—</td>
<td>7.0</td>
<td>10</td>
<td>mA</td>
</tr>
<tr>
<td>Zero Pressure Offset(^{(3)})</td>
<td>(V_{off})</td>
<td>0.088</td>
<td>0.2</td>
<td>0.313</td>
<td>Vdc</td>
</tr>
<tr>
<td>Full Scale Output(^{(4)})</td>
<td>(V_{FSO})</td>
<td>4.587</td>
<td>4.7</td>
<td>4.813</td>
<td>Vdc</td>
</tr>
<tr>
<td>Full Scale Span(^{(5)})</td>
<td>(V_{FSS})</td>
<td>—</td>
<td>4.5</td>
<td>—</td>
<td>Vdc</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>(V/P)</td>
<td>—</td>
<td>4.5</td>
<td>—</td>
<td>mV/kPa</td>
</tr>
<tr>
<td>Accuracy(^{(6)})</td>
<td>—</td>
<td>—</td>
<td>±2.5</td>
<td>%V_{FSS}</td>
<td></td>
</tr>
<tr>
<td>Response Time(^{(7)})</td>
<td>(t_R)</td>
<td>1.0</td>
<td>—</td>
<td>—</td>
<td>ms</td>
</tr>
<tr>
<td>Output Source Current at Full Scale Output</td>
<td>(I_{OS})</td>
<td>—</td>
<td>0.1</td>
<td>—</td>
<td>mA</td>
</tr>
<tr>
<td>Warm-Up Time(^{(8)})</td>
<td>—</td>
<td>20</td>
<td>—</td>
<td>—</td>
<td>ms</td>
</tr>
</tbody>
</table>

1. 1.0 kPa (kiloPascal) equals 0.145 psi.
2. Device is ratiometric within this specified excitation range.
3. Offset \((V_{off})\) is defined as the output voltage at the minimum rated pressure.
4. Full Scale Output \((V_{FSO})\) is defined as the output voltage at the maximum or full rated pressure.
5. Full Scale Span \((V_{FSS})\) is defined as the algebraic difference between the output voltage at full rated pressure and the output voltage at the minimum rated pressure.
6. Accuracy (error budget) consists of the following:
   - Linearity: Output deviation from a straight line relationship with pressure over the specified pressure range.
   - Temperature Hysteresis: Output deviation at any temperature within the operating temperature range, after the temperature is cycled to and from the minimum or maximum operating temperature points, with zero differential pressure applied.
   - Pressure Hysteresis: Output deviation at any pressure within the specified range, when this pressure is cycled to and from the minimum or maximum rated pressure, at 25°C.
   - TC Span: Output deviation over the temperature range of 0°C to 85°C, relative to 25°C.
   - TC Offset: Output deviation with minimum rated pressure applied, over the temperature range of 0°C to 85°C, relative to 25°C.
   - Variation from Nominal: The variation from nominal values, for Offset or Full Scale Span, as a percent of \(V_{FSS}\) at 25°C.
7. Response Time is defined as the time for the incremental change in the output to go from 10% to 90% of its final value when subjected to a specified step change in pressure.
8. Warm-up Time is defined as the time required for the device to meet the specified output voltage after the pressure has been stabilized.
Maximum Ratings

Table 2. Maximum Ratings\(^{(1)}\)

<table>
<thead>
<tr>
<th>Rating</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Pressure(\leq 1) Atmosphere</td>
<td>(P_{1_{\text{max}}})</td>
<td>4000</td>
<td>kPa</td>
</tr>
<tr>
<td>Storage Temperature</td>
<td>(T_{stg})</td>
<td>–40 to +125</td>
<td>°C</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>(T_A)</td>
<td>–40 to +125</td>
<td>°C</td>
</tr>
</tbody>
</table>

1. Extended exposure at the specified limits may cause permanent damage or degradation to the device.
2. This sensor is designed for applications where \(P_1\) is always greater than, or equal to \(P_2\). \(P_2\) maximum is 500 kPa.

*Figure 1 shows a block diagram of the internal circuitry integrated on the stand-alone sensing chip.*
Pressure

On-chip Temperature Compensation and Calibration

Figure 2 shows the sensor output signal relative to pressure input. Typical, minimum, and maximum output curves are shown for operation over a temperature range of 0° to 85°C using the decoupling circuit shown in Figure 4. The output will saturate outside of the specified pressure range.

The performance over temperature is achieved by integrating the shear-stress strain gauge, temperature compensation, calibration and signal conditioning circuitry onto a single monolithic chip.

Figure 3 illustrates the differential or gauge configuration in the basic chip carrier (Case 867). A fluorosilicone gel isolates the die surface and wire bonds from harsh environments, while allowing the pressure signal to be transmitted to the silicon diaphragm. The MPX5999D pressure sensor operating characteristics, and internal reliability and qualification tests are based on use of dry air as the pressure media. Media other than dry air may have adverse effects on sensor performance and long-term reliability. Contact the factory for information regarding media compatibility in your application.

Figure 4 shows the recommended decoupling circuit for interfacing the output of the integrated sensor to the A/D input of a microprocessor or microcontroller. Proper decoupling of the power supply is recommended.

Figure 2. Output vs. Pressure Differential

Figure 3. Cross-Sectional Diagrams (not to scale)

Figure 4. Recommended Power Supply Decoupling and Output Filtering
(For additional output filtering, please refer to Application Note AN1646)
Freescale designates the two sides of the pressure sensor as the Pressure (P1) side and the Vacuum (P2) side. The Pressure (P1) side is the side containing fluorosilicone gel which protects the die from harsh media. The Freescale MPX pressure sensor is designed to operate with positive differential pressure applied, P1 > P2.

The Pressure (P1) side may be identified by using the following table.

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Case Type</th>
<th>Pressure (P1) Side Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPX5999D</td>
<td>867</td>
<td>Stainless Steel Cap</td>
</tr>
</tbody>
</table>
PACKAGE DIMENSIONS

NOTES:
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION A IS INCLUSIVE OF THE MOLD STOP RING. MOLD STOP RING NOT TO EXCEED 16.00 (0.630).

CASE 867-08
ISSUE N
BASIC ELEMENT

MPX5999D
Sensors
6 Freescale Semiconductor
Pressure

INCHES MILLIMETERS

<table>
<thead>
<tr>
<th>DIM</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>F</th>
<th>G</th>
<th>J</th>
<th>L</th>
<th>M</th>
<th>N</th>
<th>R</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIN</td>
<td>0.595</td>
<td>0.514</td>
<td>0.200</td>
<td>0.027</td>
<td>0.048</td>
<td>0.014</td>
<td>0.695</td>
<td>0.475</td>
<td>0.430</td>
<td>0.090</td>
<td>0.630</td>
<td>0.534</td>
</tr>
<tr>
<td>MAX</td>
<td>0.630</td>
<td>0.534</td>
<td>0.220</td>
<td>0.033</td>
<td>0.064</td>
<td>0.016</td>
<td>0.725</td>
<td>0.495</td>
<td>0.450</td>
<td>0.105</td>
<td>0.630</td>
<td>0.534</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DIM</th>
<th></th>
<th>A</th>
<th></th>
<th></th>
<th></th>
<th>T</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MIN</td>
<td>15.11</td>
<td>13.06</td>
<td>5.08</td>
<td>0.68</td>
<td>1.22</td>
<td>0.36</td>
<td>17.65</td>
<td>12.07</td>
<td>10.92</td>
<td>2.29</td>
<td>2.66</td>
<td>205</td>
</tr>
</tbody>
</table>
16.3 General-Purpose Timers Register Manual

16.3.1 GP Timer Register Map

16.3.1.1 Instance Summary

Table 16-12 lists the base address and block size for the GP timer module instances. All timers are memory mapped to the L4 peripheral bus memory space.

Table 16-12. GP Timer Instance Summary

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Base Address</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPTIMER1</td>
<td>0x4831 8000</td>
<td>4K bytes</td>
</tr>
<tr>
<td>GPTIMER2</td>
<td>0x4903 2000</td>
<td>4K bytes</td>
</tr>
<tr>
<td>GPTIMER3</td>
<td>0x4903 4000</td>
<td>4K bytes</td>
</tr>
<tr>
<td>GPTIMER4</td>
<td>0x4903 6000</td>
<td>4K bytes</td>
</tr>
<tr>
<td>GPTIMER5</td>
<td>0x4903 8000</td>
<td>4K bytes</td>
</tr>
<tr>
<td>GPTIMER6</td>
<td>0x4903 A000</td>
<td>4K bytes</td>
</tr>
<tr>
<td>GPTIMER7</td>
<td>0x4903 C000</td>
<td>4K bytes</td>
</tr>
<tr>
<td>GPTIMER8</td>
<td>0x4903 E000</td>
<td>4K bytes</td>
</tr>
<tr>
<td>GPTIMER9</td>
<td>0x4904 0000</td>
<td>4K bytes</td>
</tr>
<tr>
<td>GPTIMER10</td>
<td>0x4808 6000</td>
<td>4K bytes</td>
</tr>
<tr>
<td>GPTIMER11</td>
<td>0x4808 8000</td>
<td>4K bytes</td>
</tr>
</tbody>
</table>

16.3.2 GP Timer Register Mapping Summary

CAUTION
The GP timer registers are limited to 32-bit and 16-bit data accesses; 8-bit access is not allowed and can corrupt the register content.

Table 16-13 through Table 16-15 provide the register summary and associated offset addresses for the 11 GP timer internal registers. (Example: The physical address for the TCLR register of GPTIMER8 is 0x4903 E024.)
<table>
<thead>
<tr>
<th>Register Name</th>
<th>Type</th>
<th>Width (Bits)</th>
<th>Address Offset</th>
<th>Physical Address (GPTIMER1)</th>
<th>Physical Address (GPTIMER2)</th>
<th>Physical Address (GPTIMER3)</th>
<th>Physical Address (GPTIMER4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIDR</td>
<td>R</td>
<td>32</td>
<td>0x000</td>
<td>0x4831 8000</td>
<td>0x4903 2000</td>
<td>0x4903 4000</td>
<td>0x4903 6000</td>
</tr>
<tr>
<td>TIOCP_CFG</td>
<td>RW</td>
<td>32</td>
<td>0x010</td>
<td>0x4831 8010</td>
<td>0x4903 2010</td>
<td>0x4903 4010</td>
<td>0x4903 6010</td>
</tr>
<tr>
<td>TISTAT</td>
<td>R</td>
<td>32</td>
<td>0x014</td>
<td>0x4831 8014</td>
<td>0x4903 2014</td>
<td>0x4903 4014</td>
<td>0x4903 6014</td>
</tr>
<tr>
<td>TISR</td>
<td>RW</td>
<td>32</td>
<td>0x018</td>
<td>0x4831 8018</td>
<td>0x4903 2018</td>
<td>0x4903 4018</td>
<td>0x4903 6018</td>
</tr>
<tr>
<td>TIER</td>
<td>RW</td>
<td>32</td>
<td>0x01C</td>
<td>0x4831 801C</td>
<td>0x4903 201C</td>
<td>0x4903 401C</td>
<td>0x4903 601C</td>
</tr>
<tr>
<td>TWER</td>
<td>RW</td>
<td>32</td>
<td>0x020</td>
<td>0x4831 8020</td>
<td>0x4903 2020</td>
<td>0x4903 4020</td>
<td>0x4903 6020</td>
</tr>
<tr>
<td>TCLR</td>
<td>RW</td>
<td>32</td>
<td>0x024</td>
<td>0x4831 8024</td>
<td>0x4903 2024</td>
<td>0x4903 4024</td>
<td>0x4903 6024</td>
</tr>
<tr>
<td>TCRR</td>
<td>RW</td>
<td>32</td>
<td>0x028</td>
<td>0x4831 8028</td>
<td>0x4903 2028</td>
<td>0x4903 4028</td>
<td>0x4903 6028</td>
</tr>
<tr>
<td>TLDR</td>
<td>RW</td>
<td>32</td>
<td>0x02C</td>
<td>0x4831 802C</td>
<td>0x4903 202C</td>
<td>0x4903 402C</td>
<td>0x4903 602C</td>
</tr>
<tr>
<td>TTGR</td>
<td>RW</td>
<td>32</td>
<td>0x030</td>
<td>0x4831 8030</td>
<td>0x4903 2030</td>
<td>0x4903 4030</td>
<td>0x4903 6030</td>
</tr>
<tr>
<td>TWPS</td>
<td>R</td>
<td>32</td>
<td>0x034</td>
<td>0x4831 8034</td>
<td>0x4903 2034</td>
<td>0x4903 4034</td>
<td>0x4903 6034</td>
</tr>
<tr>
<td>TMAR</td>
<td>RW</td>
<td>32</td>
<td>0x038</td>
<td>0x4831 8038</td>
<td>0x4903 2038</td>
<td>0x4903 4038</td>
<td>0x4903 6038</td>
</tr>
<tr>
<td>TCBR1</td>
<td>R</td>
<td>32</td>
<td>0x03C</td>
<td>0x4831 803C</td>
<td>0x4903 203C</td>
<td>0x4903 403C</td>
<td>0x4903 603C</td>
</tr>
<tr>
<td>TCBR2</td>
<td>RW</td>
<td>32</td>
<td>0x040</td>
<td>0x4831 8040</td>
<td>0x4903 2040</td>
<td>0x4903 4040</td>
<td>0x4903 6040</td>
</tr>
<tr>
<td>TCBR3</td>
<td>RW</td>
<td>32</td>
<td>0x044</td>
<td>0x4831 8044</td>
<td>0x4903 2044</td>
<td>0x4903 4044</td>
<td>0x4903 6044</td>
</tr>
<tr>
<td>TPIR</td>
<td>RW</td>
<td>32</td>
<td>0x048</td>
<td>0x4831 8048</td>
<td>0x4903 2048</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TNIR</td>
<td>RW</td>
<td>32</td>
<td>0x04C</td>
<td>0x4831 804C</td>
<td>0x4903 204C</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TCVR</td>
<td>RW</td>
<td>32</td>
<td>0x050</td>
<td>0x4831 8050</td>
<td>0x4903 2050</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TOCR</td>
<td>RW</td>
<td>32</td>
<td>0x054</td>
<td>0x4831 8054</td>
<td>0x4903 2054</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TOWR</td>
<td>RW</td>
<td>32</td>
<td>0x058</td>
<td>0x4831 8058</td>
<td>0x4903 2058</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Register Name</td>
<td>Type</td>
<td>Register Width (Bits)</td>
<td>Address Offset</td>
<td>Physical Address (GPTIMER5)</td>
<td>Physical Address (GPTIMER6)</td>
<td>Physical Address (GPTIMER7)</td>
<td>Physical Address (GPTIMER8)</td>
</tr>
<tr>
<td>---------------</td>
<td>------</td>
<td>----------------------</td>
<td>----------------</td>
<td>---------------------------</td>
<td>---------------------------</td>
<td>---------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>TIDR</td>
<td>R</td>
<td>32</td>
<td>0x000</td>
<td>0x4903 8000</td>
<td>0x4903 A000</td>
<td>0x4903 C000</td>
<td>0x4903 E000</td>
</tr>
<tr>
<td>TIOCP_CFG</td>
<td>RW</td>
<td>32</td>
<td>0x010</td>
<td>0x4903 8010</td>
<td>0x4903 A010</td>
<td>0x4903 C010</td>
<td>0x4903 E010</td>
</tr>
<tr>
<td>TISTAT</td>
<td>R</td>
<td>32</td>
<td>0x014</td>
<td>0x4903 8014</td>
<td>0x4903 A014</td>
<td>0x4903 C014</td>
<td>0x4903 E014</td>
</tr>
<tr>
<td>TISR</td>
<td>RW</td>
<td>32</td>
<td>0x018</td>
<td>0x4903 8018</td>
<td>0x4903 A018</td>
<td>0x4903 C018</td>
<td>0x4903 E018</td>
</tr>
<tr>
<td>TIER</td>
<td>RW</td>
<td>32</td>
<td>0x01C</td>
<td>0x4903 801C</td>
<td>0x4903 A01C</td>
<td>0x4903 C01C</td>
<td>0x4903 E01C</td>
</tr>
<tr>
<td>TWER</td>
<td>RW</td>
<td>32</td>
<td>0x020</td>
<td>0x4903 8020</td>
<td>0x4903 A020</td>
<td>0x4903 C020</td>
<td>0x4903 E020</td>
</tr>
<tr>
<td>TCLR</td>
<td>RW</td>
<td>32</td>
<td>0x024</td>
<td>0x4903 8024</td>
<td>0x4903 A024</td>
<td>0x4903 C024</td>
<td>0x4903 E024</td>
</tr>
<tr>
<td>TCRR</td>
<td>RW</td>
<td>32</td>
<td>0x028</td>
<td>0x4903 8028</td>
<td>0x4903 A028</td>
<td>0x4903 C028</td>
<td>0x4903 E028</td>
</tr>
<tr>
<td>TLDR</td>
<td>RW</td>
<td>32</td>
<td>0x02C</td>
<td>0x4903 802C</td>
<td>0x4903 A02C</td>
<td>0x4903 C02C</td>
<td>0x4903 E02C</td>
</tr>
<tr>
<td>TTGR</td>
<td>RW</td>
<td>32</td>
<td>0x030</td>
<td>0x4903 8030</td>
<td>0x4903 A030</td>
<td>0x4903 C030</td>
<td>0x4903 E030</td>
</tr>
<tr>
<td>TWPS</td>
<td>R</td>
<td>32</td>
<td>0x034</td>
<td>0x4903 8034</td>
<td>0x4903 A034</td>
<td>0x4903 C034</td>
<td>0x4903 E034</td>
</tr>
<tr>
<td>TMAR</td>
<td>RW</td>
<td>32</td>
<td>0x038</td>
<td>0x4903 8038</td>
<td>0x4903 A038</td>
<td>0x4903 C038</td>
<td>0x4903 E038</td>
</tr>
<tr>
<td>TSC1R</td>
<td>R</td>
<td>32</td>
<td>0x03C</td>
<td>0x4903 803C</td>
<td>0x4903 A03C</td>
<td>0x4903 C03C</td>
<td>0x4903 E03C</td>
</tr>
<tr>
<td>TSC1R</td>
<td>RW</td>
<td>32</td>
<td>0x040</td>
<td>0x4903 8040</td>
<td>0x4903 A040</td>
<td>0x4903 C040</td>
<td>0x4903 E040</td>
</tr>
<tr>
<td>TSC1R</td>
<td>R</td>
<td>32</td>
<td>0x044</td>
<td>0x4903 8044</td>
<td>0x4903 A044</td>
<td>0x4903 C044</td>
<td>0x4903 E044</td>
</tr>
</tbody>
</table>
Table 16-15. GPTIMER9 to GPTIMER11 Register Summary

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Type</th>
<th>Width (Bits)</th>
<th>Address Offset</th>
<th>Physical Address (GPTIMER9)</th>
<th>Physical Address (GPTIMER10)</th>
<th>Physical Address (GPTIMER11)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIDR</td>
<td>R</td>
<td>32</td>
<td>0x000</td>
<td>0x4904 0000</td>
<td>0x4808 6000</td>
<td>0x4808 8000</td>
</tr>
<tr>
<td>TIOCP_CFG</td>
<td>RW</td>
<td>32</td>
<td>0x010</td>
<td>0x4904 0010</td>
<td>0x4808 6010</td>
<td>0x4808 8010</td>
</tr>
<tr>
<td>TISTAT</td>
<td>R</td>
<td>32</td>
<td>0x014</td>
<td>0x4904 0014</td>
<td>0x4808 6014</td>
<td>0x4808 8014</td>
</tr>
<tr>
<td>TISR</td>
<td>RW</td>
<td>32</td>
<td>0x018</td>
<td>0x4904 0018</td>
<td>0x4808 6018</td>
<td>0x4808 8018</td>
</tr>
<tr>
<td>TIER</td>
<td>RW</td>
<td>32</td>
<td>0x020</td>
<td>0x4904 0020</td>
<td>0x4808 6020</td>
<td>0x4808 8020</td>
</tr>
<tr>
<td>TWER</td>
<td>RW</td>
<td>32</td>
<td>0x024</td>
<td>0x4904 0024</td>
<td>0x4808 6024</td>
<td>0x4808 8024</td>
</tr>
<tr>
<td>TCLR</td>
<td>RW</td>
<td>32</td>
<td>0x028</td>
<td>0x4904 0028</td>
<td>0x4808 6028</td>
<td>0x4808 8028</td>
</tr>
<tr>
<td>TLDR</td>
<td>RW</td>
<td>32</td>
<td>0x02C</td>
<td>0x4904 002C</td>
<td>0x4808 602C</td>
<td>0x4808 802C</td>
</tr>
<tr>
<td>TTGR</td>
<td>RW</td>
<td>32</td>
<td>0x030</td>
<td>0x4904 0030</td>
<td>0x4808 6030</td>
<td>0x4808 8030</td>
</tr>
<tr>
<td>TWPS</td>
<td>R</td>
<td>32</td>
<td>0x034</td>
<td>0x4904 0034</td>
<td>0x4808 6034</td>
<td>0x4808 8034</td>
</tr>
<tr>
<td>TMAR</td>
<td>RW</td>
<td>32</td>
<td>0x038</td>
<td>0x4904 0038</td>
<td>0x4808 6038</td>
<td>0x4808 8038</td>
</tr>
<tr>
<td>TGCAR1</td>
<td>R</td>
<td>32</td>
<td>0x03C</td>
<td>0x4904 003C</td>
<td>0x4808 603C</td>
<td>0x4808 803C</td>
</tr>
<tr>
<td>TSCICR</td>
<td>RW</td>
<td>32</td>
<td>0x040</td>
<td>0x4904 0040</td>
<td>0x4808 6040</td>
<td>0x4808 8040</td>
</tr>
<tr>
<td>TGCAR2</td>
<td>R</td>
<td>32</td>
<td>0x044</td>
<td>0x4904 0044</td>
<td>0x4808 6044</td>
<td>0x4808 8044</td>
</tr>
<tr>
<td>TPIR</td>
<td>RW</td>
<td>32</td>
<td>0x048</td>
<td>-</td>
<td>0x4808 6048</td>
<td>-</td>
</tr>
<tr>
<td>TNIR</td>
<td>RW</td>
<td>32</td>
<td>0x04C</td>
<td>-</td>
<td>0x4808 604C</td>
<td>-</td>
</tr>
<tr>
<td>TCVR</td>
<td>RW</td>
<td>32</td>
<td>0x050</td>
<td>-</td>
<td>0x4808 6050</td>
<td>-</td>
</tr>
<tr>
<td>TOCR</td>
<td>RW</td>
<td>32</td>
<td>0x054</td>
<td>-</td>
<td>0x4808 6054</td>
<td>-</td>
</tr>
<tr>
<td>TOWR</td>
<td>RW</td>
<td>32</td>
<td>0x058</td>
<td>-</td>
<td>0x4808 6058</td>
<td>-</td>
</tr>
</tbody>
</table>

16.3.3  GP Timer Register Descriptions

Table 16-16 through Table 16-54 describe the GP timer register bits.
## Table 16-16. TIDR

<table>
<thead>
<tr>
<th>Address Offset</th>
<th>Physical Address</th>
<th>Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x000</td>
<td>0x4831 8000</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td></td>
<td>0x4903 2000</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td></td>
<td>0x4903 4000</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td></td>
<td>0x4903 6000</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td></td>
<td>0x4903 8000</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td></td>
<td>0x4903 A000</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td></td>
<td>0x4903 C000</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td></td>
<td>0x4903 E000</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td></td>
<td>0x4904 0000</td>
<td>GPTIMER9</td>
</tr>
<tr>
<td></td>
<td>0x4808 6000</td>
<td>GPTIMER9</td>
</tr>
<tr>
<td></td>
<td>0x4808 8000</td>
<td>GPTIMER9</td>
</tr>
<tr>
<td></td>
<td>0x4903 2000</td>
<td>GPT2</td>
</tr>
<tr>
<td></td>
<td>0x4903 4000</td>
<td>GPT3</td>
</tr>
<tr>
<td></td>
<td>0x4903 6000</td>
<td>GPT4</td>
</tr>
<tr>
<td></td>
<td>0x4903 8000</td>
<td>GPT5</td>
</tr>
<tr>
<td></td>
<td>0x4903 A000</td>
<td>GPT6</td>
</tr>
<tr>
<td></td>
<td>0x4903 C000</td>
<td>GPT7</td>
</tr>
<tr>
<td></td>
<td>0x4903 E000</td>
<td>GPT8</td>
</tr>
<tr>
<td></td>
<td>0x4904 0000</td>
<td>GPT9</td>
</tr>
<tr>
<td></td>
<td>0x4808 6000</td>
<td>GPT10</td>
</tr>
<tr>
<td></td>
<td>0x4808 8000</td>
<td>GPT11</td>
</tr>
</tbody>
</table>

### Description
This register contains the IP revision code.

### Type
R

<table>
<thead>
<tr>
<th>Bits</th>
<th>Field Name</th>
<th>Description</th>
<th>Type</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:8</td>
<td>Reserved</td>
<td>Reads return 0.</td>
<td>R</td>
<td>0x000000</td>
</tr>
<tr>
<td>7:0</td>
<td>TID_REV</td>
<td>IP revision</td>
<td>R</td>
<td>See (1)</td>
</tr>
</tbody>
</table>

### Notes
(1) TI internal data

### Table 16-17. Register Call Summary for Register TIDR

- General-Purpose Timers
  - Accessing GP Timer Registers: [0]

- General-Purpose Timers Register Manual
  - GP Timer Register Mapping Summary: [1] [2] [3]
### Table 16-27. Register Call Summary for Register TWER

**General-Purpose Timers**
- Wake-Up Capability: [0] [1]
- GP Timers Functional Description: [2]
- Accessing GP Timer Registers: [3]
- Writing to Timer Registers: [4]

**General-Purpose Timers Register Manual**
- GP Timer Register Mapping Summary: [5] [6] [7]

#### Table 16-28. TCLR

<table>
<thead>
<tr>
<th>Address Offset</th>
<th>Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x024</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td>0x4837 8024</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td>0x4903 2024</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td>0x4903 4024</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td>0x4903 6024</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td>0x4903 8024</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td>0x4903 A024</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td>0x4903 C024</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td>0x4903 E024</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td>0x4904 0024</td>
<td>GPTIMER9</td>
</tr>
<tr>
<td>0x4808 6024</td>
<td>GPTIMER9</td>
</tr>
<tr>
<td>0x4808 8024</td>
<td>GPTIMER9</td>
</tr>
<tr>
<td>0x4903 2024</td>
<td>GPT2</td>
</tr>
<tr>
<td>0x4903 4024</td>
<td>GPT3</td>
</tr>
<tr>
<td>0x4903 6024</td>
<td>GPT4</td>
</tr>
<tr>
<td>0x4903 8024</td>
<td>GPT5</td>
</tr>
<tr>
<td>0x4903 A024</td>
<td>GPT6</td>
</tr>
<tr>
<td>0x4903 C024</td>
<td>GPT7</td>
</tr>
<tr>
<td>0x4903 E024</td>
<td>GPT8</td>
</tr>
<tr>
<td>0x4904 0024</td>
<td>GPT9</td>
</tr>
<tr>
<td>0x4808 6024</td>
<td>GPT10</td>
</tr>
<tr>
<td>0x4808 8024</td>
<td>GPT11</td>
</tr>
</tbody>
</table>

**Description**
This register controls optional features specific to the timer functionality.

**Type**
RW

---

<table>
<thead>
<tr>
<th>Bits Field Name</th>
<th>Description</th>
<th>Type</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 OVF_WUP_ENA</td>
<td>Enable overflow wake-up</td>
<td>RW</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0x0: Disable overflow wake-up.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x1: Enable overflow wake-up.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 MAT_WUP_ENA</td>
<td>Enable match wake-up</td>
<td>RW</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0x0: Disable match wake-up.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x1: Enable match wake-up.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bits</td>
<td>Field Name</td>
<td>Description</td>
<td>Type</td>
</tr>
<tr>
<td>-----------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>31:15</td>
<td>Reserved</td>
<td>Reads return 0.</td>
<td>R</td>
</tr>
<tr>
<td>14</td>
<td>GPO_CFG</td>
<td>PWM output/event detection input pin direction control:</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: Configures the pin as an output (needed when PWM mode is required)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: Configures the pin as an input (needed when capture mode is required)</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>CAPT_MODE</td>
<td>Capture mode select bit (first/second)</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: Capture the first enabled capture event in TCAR1.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: Capture the second enabled capture event in TCAR2.</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>PT</td>
<td>Pulse or toggle select bit</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: Pulse modulation</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: Toggle modulation</td>
<td></td>
</tr>
<tr>
<td>11:10</td>
<td>TRG</td>
<td>Trigger output mode</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: No trigger</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: Overflow trigger</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x2: Overflow and match trigger</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x3: Reserved</td>
<td></td>
</tr>
<tr>
<td>9:8</td>
<td>TCM</td>
<td>Transition capture mode</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: No capture</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: Capture on rising edges of EVENT_CAPTURE pin.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x2: Capture on falling edges of EVENT_CAPTURE pin.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x3: Capture on both edges of EVENT_CAPTURE pin.</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>SCPWM</td>
<td>Pulse-width-modulation output pin default setting when counter is stopped or trigger output mode is set to no trigger.</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: Default value of PWM_out output: 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: Default value of PWM_out output: 1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>CE</td>
<td>Compare enable</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: Compare disabled</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: Compare enabled</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>PRE</td>
<td>Prescaler enable</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: Prescaler disabled</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: Prescaler enabled</td>
<td></td>
</tr>
<tr>
<td>4:2</td>
<td>PTV</td>
<td>Trigger output mode</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: The timer counter is prescaled with the value: $2^{PTV+1}$. Example: PTV = 3, counter increases value if started after 16 functional clock periods.</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>AR</td>
<td>Autoreload mode</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: One-shot mode overflow</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: Autoreload mode overflow</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>ST</td>
<td>Start/stop timer control</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0: Stop the timer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x1: Start the timer</td>
<td></td>
</tr>
<tr>
<td>Bits</td>
<td>Field Name</td>
<td>Description</td>
<td>Type</td>
</tr>
<tr>
<td>------</td>
<td>----------------</td>
<td>------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>31:0</td>
<td>TIMER_COUNTER</td>
<td>The value of the timer counter register</td>
<td>RW</td>
</tr>
</tbody>
</table>

Table 16-29. Register Call Summary for Register TCLR

- Timer Mode Functionality: [0] [1] [2] [3] [4] [5] [6]
- Compare Mode Functionality: [16] [17] [18]
- Prescaler Functionality: [19] [20]
- Pulse-Width Modulation: [21] [22] [23] [24] [25] [26] [27] [28] [29] [30]
- Timer Counting Rate: [31] [32] [33] [34] [35] [36] [37]
- Accessing GP Timer Registers: [38]
- Writing to Timer Registers: [39]
- Write Posting Synchronization Mode: [40]

Table 16-30. TCRR

- GP Timer Register Mapping Summary: [41] [42] [43] [44]

<table>
<thead>
<tr>
<th>Address Offset</th>
<th>Physical Address</th>
<th>Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x028</td>
<td>0x4831 8028</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td>0x4903 2028</td>
<td>GPTIMER1</td>
<td></td>
</tr>
<tr>
<td>0x4903 4028</td>
<td>GPTIMER1</td>
<td></td>
</tr>
<tr>
<td>0x4903 6028</td>
<td>GPTIMER1</td>
<td></td>
</tr>
<tr>
<td>0x4903 8028</td>
<td>GPTIMER5</td>
<td></td>
</tr>
<tr>
<td>0x4903 A028</td>
<td>GPTIMER5</td>
<td></td>
</tr>
<tr>
<td>0x4903 C028</td>
<td>GPTIMER5</td>
<td></td>
</tr>
<tr>
<td>0x4904 E028</td>
<td>GPTIMER9</td>
<td></td>
</tr>
<tr>
<td>0x4806 6028</td>
<td>GPTIMER9</td>
<td></td>
</tr>
<tr>
<td>0x4806 8028</td>
<td>GPTIMER9</td>
<td></td>
</tr>
<tr>
<td>0x4903 2028</td>
<td>GPT2</td>
<td></td>
</tr>
<tr>
<td>0x4903 4028</td>
<td>GPT3</td>
<td></td>
</tr>
<tr>
<td>0x4903 6028</td>
<td>GPT4</td>
<td></td>
</tr>
<tr>
<td>0x4903 8028</td>
<td>GPT5</td>
<td></td>
</tr>
<tr>
<td>0x4903 A028</td>
<td>GPT6</td>
<td></td>
</tr>
<tr>
<td>0x4903 C028</td>
<td>GPT7</td>
<td></td>
</tr>
<tr>
<td>0x4903 E028</td>
<td>GPT8</td>
<td></td>
</tr>
<tr>
<td>0x4904 0028</td>
<td>GPT9</td>
<td></td>
</tr>
<tr>
<td>0x4806 6028</td>
<td>GPT10</td>
<td></td>
</tr>
<tr>
<td>0x4806 8028</td>
<td>GPT11</td>
<td></td>
</tr>
</tbody>
</table>

Description: This register holds the value of the internal counter.

Type: RW
Table 16-31. Register Call Summary for Register TCRR

<table>
<thead>
<tr>
<th>General-Purpose Timers</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Wake-Up Capability: [0] [1]</td>
</tr>
<tr>
<td>• 1-ms Tick Generation (Only GPTIMER1, GPTIMER2, and GPTIMER10): [11] [12] [13]</td>
</tr>
<tr>
<td>• Capture Mode Functionality: [14] [15]</td>
</tr>
<tr>
<td>• Compare Mode Functionality: [16] [17]</td>
</tr>
<tr>
<td>• Prescaler Functionality: [18] [19]</td>
</tr>
<tr>
<td>• Accessing GP Timer Registers: [20]</td>
</tr>
<tr>
<td>• Writing to Timer Registers: [21]</td>
</tr>
<tr>
<td>• Write Posting Synchronization Mode: [22] [23]</td>
</tr>
<tr>
<td>• Reading From Timer Counter Registers: [24]</td>
</tr>
</tbody>
</table>

Table 16-32. TLDR

<table>
<thead>
<tr>
<th>Address Offset</th>
<th>Physical Address</th>
<th>Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02C</td>
<td>0x4831 802C</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td></td>
<td>0x4903 202C</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td></td>
<td>0x4903 402C</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td></td>
<td>0x4903 602C</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td></td>
<td>0x4903 802C</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td></td>
<td>0x4903 A02C</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td></td>
<td>0x4903 C02C</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td></td>
<td>0x4903 E02C</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td></td>
<td>0x4904 002C</td>
<td>GPTIMERS</td>
</tr>
<tr>
<td></td>
<td>0x4908 602C</td>
<td>GPTIMER9</td>
</tr>
<tr>
<td></td>
<td>0x4908 802C</td>
<td>GPTIMER9</td>
</tr>
<tr>
<td></td>
<td>0x4903 202C</td>
<td>GPT2</td>
</tr>
<tr>
<td></td>
<td>0x4903 402C</td>
<td>GPT3</td>
</tr>
<tr>
<td></td>
<td>0x4903 602C</td>
<td>GPT4</td>
</tr>
<tr>
<td></td>
<td>0x4903 802C</td>
<td>GPT5</td>
</tr>
<tr>
<td></td>
<td>0x4903 A02C</td>
<td>GPT6</td>
</tr>
<tr>
<td></td>
<td>0x4903 C02C</td>
<td>GPT7</td>
</tr>
<tr>
<td></td>
<td>0x4903 E02C</td>
<td>GPT8</td>
</tr>
<tr>
<td></td>
<td>0x4904 002C</td>
<td>GPT9</td>
</tr>
<tr>
<td></td>
<td>0x4908 602C</td>
<td>GPT10</td>
</tr>
<tr>
<td></td>
<td>0x4908 802C</td>
<td>GPT11</td>
</tr>
</tbody>
</table>

Description: This register holds the timer load values.

Type: RW

<table>
<thead>
<tr>
<th>Bits</th>
<th>Field Name</th>
<th>Description</th>
<th>Type</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>31.0</td>
<td>LOAD_VALUE</td>
<td>The value of the timer load register</td>
<td>RW</td>
<td>0x00000000</td>
</tr>
</tbody>
</table>
### Table 16-33. Register Call Summary for Register TLDR

**General-Purpose Timers**
- Timer Mode Functionality: [0] [1] [2]
- Prescaler Functionality: [13] [14]
- Pulse-Width Modulation: [15] [16] [17]
- Timer Counting Rate: [18] [19] [20] [21] [22]
- Accessing GP Timer Registers: [23]
- Writing to Timer Registers: [24]
- Write Posting Synchronization Mode: [25]

**General-Purpose Timers Register Manual**
- GP Timer Register Mapping Summary: [26] [27] [28]

### Table 16-34. TTGR

<table>
<thead>
<tr>
<th>Address Offset</th>
<th>Physical Address</th>
<th>Instance</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x030</td>
<td>0x4831 08030</td>
<td>GPTIMER1</td>
<td>This register triggers a counter reload of timer by writing any value in it.</td>
</tr>
<tr>
<td></td>
<td>0x4903 02030</td>
<td>GPTIMER1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 04030</td>
<td>GPTIMER1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 06030</td>
<td>GPTIMER5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 08030</td>
<td>GPTIMER5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 A030</td>
<td>GPTIMER9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4906 08030</td>
<td>GPTIMER9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 2030</td>
<td>GPT2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 4030</td>
<td>GPT3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 6030</td>
<td>GPT4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 8030</td>
<td>GPT5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 A030</td>
<td>GPT6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 C030</td>
<td>GPT7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4903 E030</td>
<td>GPT8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4904 0030</td>
<td>GPT9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4808 6030</td>
<td>GPT10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x4808 8030</td>
<td>GPT11</td>
<td></td>
</tr>
</tbody>
</table>

**Type**: RW

<table>
<thead>
<tr>
<th>Bits</th>
<th>Field Name</th>
<th>Description</th>
<th>Type</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>TTGR_ VALUE</td>
<td>The value of the trigger register. During reads, it always returns 0xFFFFFFFF.</td>
<td>RW</td>
<td>0xFFFFFFFF</td>
</tr>
</tbody>
</table>

### Table 16-35. Register Call Summary for Register TTGR

**General-Purpose Timers**
- Timer Mode Functionality: [0] [1]
- Prescaler Functionality: [2]
- Accessing GP Timer Registers: [3]
- Writing to Timer Registers: [4]
- Write Posting Synchronization Mode: [5]

**General-Purpose Timers Register Manual**
- GP Timer Register Mapping Summary: [6] [7] [8]

Submit Documentation Feedback

© 2010–2011, Texas Instruments Incorporated
Table 16-38. TMAR

<table>
<thead>
<tr>
<th>Address Offset</th>
<th>Physical Address</th>
<th>Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x038</td>
<td>0x4831 8038</td>
<td>GPTIMER1</td>
</tr>
<tr>
<td>0x4903 2038</td>
<td>GPTIMER1</td>
<td></td>
</tr>
<tr>
<td>0x4903 4038</td>
<td>GPTIMER1</td>
<td></td>
</tr>
<tr>
<td>0x4903 6038</td>
<td>GPTIMER1</td>
<td></td>
</tr>
<tr>
<td>0x4903 8038</td>
<td>GPTIMER1</td>
<td></td>
</tr>
<tr>
<td>0x4903 A038</td>
<td>GPTIMER5</td>
<td></td>
</tr>
<tr>
<td>0x4903 C038</td>
<td>GPTIMER5</td>
<td></td>
</tr>
<tr>
<td>0x4903 E038</td>
<td>GPTIMER5</td>
<td></td>
</tr>
<tr>
<td>0x4904 0038</td>
<td>GPTIMER9</td>
<td></td>
</tr>
<tr>
<td>0x4808 6038</td>
<td>GPTIMER9</td>
<td></td>
</tr>
<tr>
<td>0x4808 8038</td>
<td>GPTIMER9</td>
<td></td>
</tr>
<tr>
<td>0x4903 2038</td>
<td>GPT2</td>
<td></td>
</tr>
<tr>
<td>0x4903 4038</td>
<td>GPT3</td>
<td></td>
</tr>
<tr>
<td>0x4903 6038</td>
<td>GPT4</td>
<td></td>
</tr>
<tr>
<td>0x4903 8038</td>
<td>GPT5</td>
<td></td>
</tr>
<tr>
<td>0x4903 A038</td>
<td>GPT6</td>
<td></td>
</tr>
<tr>
<td>0x4903 C038</td>
<td>GPT7</td>
<td></td>
</tr>
<tr>
<td>0x4903 E038</td>
<td>GPT8</td>
<td></td>
</tr>
<tr>
<td>0x4904 0038</td>
<td>GPT9</td>
<td></td>
</tr>
<tr>
<td>0x4808 6038</td>
<td>GPT10</td>
<td></td>
</tr>
<tr>
<td>0x4808 8038</td>
<td>GPT11</td>
<td></td>
</tr>
</tbody>
</table>

Description
This register holds the value to be compared with the counter value.

Type
RW

<table>
<thead>
<tr>
<th>Bits</th>
<th>Field Name</th>
<th>Description</th>
<th>Type</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>COMPARE_ VALUE</td>
<td>The value of the match register</td>
<td>RW</td>
<td>0x00000000</td>
</tr>
</tbody>
</table>

Table 16-39. Register Call Summary for Register TMAR

General-Purpose Timers
- Wake-Up Capability: [0]
- Compare Mode Functionality: [1] [2] [3]
- Accessing GP Timer Registers: [7]
- Writing to Timer Registers: [8]
- Write Posting Synchronization Mode: [9]

General-Purpose Timers Register Manual
- GP Timer Register Mapping Summary: [10] [11] [12]
<table>
<thead>
<tr>
<th>REGISTER NAME</th>
<th>Pad Name</th>
<th>Physical Address</th>
<th>WakeUpX</th>
<th>OffMode</th>
<th>Input Enable</th>
<th>Reserved</th>
<th>PU/PD</th>
<th>MuxMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTROL_PADCONF_SDRC_D0[15:0]</td>
<td>sdrc_d0</td>
<td>0x4800 2030</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D0[31:16]</td>
<td>sdrc_d1</td>
<td>0x4800 2030</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D2[15:0]</td>
<td>sdrc_d2</td>
<td>0x4800 2034</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D2[31:16]</td>
<td>sdrc_d3</td>
<td>0x4800 2034</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D4[15:0]</td>
<td>sdrc_d4</td>
<td>0x4800 2038</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D4[31:16]</td>
<td>sdrc_d5</td>
<td>0x4800 2038</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D6[15:0]</td>
<td>sdrc_d6</td>
<td>0x4800 203C</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D6[31:16]</td>
<td>sdrc_d7</td>
<td>0x4800 203C</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D8[15:0]</td>
<td>sdrc_d8</td>
<td>0x4800 2040</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D8[31:16]</td>
<td>sdrc_d9</td>
<td>0x4800 2040</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D10[15:0]</td>
<td>sdrc_d10</td>
<td>0x4800 2044</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D10[31:16]</td>
<td>sdrc_d11</td>
<td>0x4800 2044</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D12[15:0]</td>
<td>sdrc_d12</td>
<td>0x4800 2048</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D12[31:16]</td>
<td>sdrc_d13</td>
<td>0x4800 2048</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D14[15:0]</td>
<td>sdrc_d14</td>
<td>0x4800 204C</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D14[31:16]</td>
<td>sdrc_d15</td>
<td>0x4800 204C</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D16[15:0]</td>
<td>sdrc_d16</td>
<td>0x4800 2050</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D16[31:16]</td>
<td>sdrc_d17</td>
<td>0x4800 2050</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D18[15:0]</td>
<td>sdrc_d18</td>
<td>0x4800 2054</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D18[31:16]</td>
<td>sdrc_d19</td>
<td>0x4800 2054</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D20[15:0]</td>
<td>sdrc_d20</td>
<td>0x4800 2058</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D20[31:16]</td>
<td>sdrc_d21</td>
<td>0x4800 2058</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D22[15:0]</td>
<td>sdrc_d22</td>
<td>0x4800 205C</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D22[31:16]</td>
<td>sdrc_d23</td>
<td>0x4800 205C</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D24[15:0]</td>
<td>sdrc_d24</td>
<td>0x4800 2060</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D24[31:16]</td>
<td>sdrc_d25</td>
<td>0x4800 2060</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D26[15:0]</td>
<td>sdrc_d26</td>
<td>0x4800 2064</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D26[31:16]</td>
<td>sdrc_d27</td>
<td>0x4800 2064</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D28[15:0]</td>
<td>sdrc_d28</td>
<td>0x4800 2068</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D28[31:16]</td>
<td>sdrc_d29</td>
<td>0x4800 2068</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D30[15:0]</td>
<td>sdrc_d30</td>
<td>0x4800 206C</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_D30[31:16]</td>
<td>sdrc_d31</td>
<td>0x4800 206C</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[15:0]</td>
<td>sdrc_clk</td>
<td>0x4800 2070</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b00</td>
<td>0b00</td>
<td>--</td>
</tr>
</tbody>
</table>
Table 7-76. CONTROL_PADCONF_CAPABILITIES (continued)

<table>
<thead>
<tr>
<th>REGISTER NAME</th>
<th>Pad Name</th>
<th>Physical Address</th>
<th>WakeUp</th>
<th>OffMode</th>
<th>Input</th>
<th>PU/PD</th>
<th>Reserved</th>
<th>MuxMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>sdrc_dqs0</td>
<td>0x4800 2070</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b000</td>
<td>0b000</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>sdrc_dqs1</td>
<td>0x4800 2074</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b000</td>
<td>0b000</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>sdrc_dqs2</td>
<td>0x4800 2078</td>
<td>--</td>
<td>--</td>
<td>0b1</td>
<td>0b000</td>
<td>0b000</td>
<td>--</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_a1</td>
<td>0x4800 2078</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_a2</td>
<td>0x4800 207C</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_a3</td>
<td>0x4800 2080</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_a4</td>
<td>0x4800 2084</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_a5</td>
<td>0x4800 2088</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_a6</td>
<td>0x4800 2092</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_a7</td>
<td>0x4800 2094</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_a8</td>
<td>0x4800 2098</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_a9</td>
<td>0x4800 209C</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_a10</td>
<td>0x4800 2000</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d0</td>
<td>0x4800 20A0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d1</td>
<td>0x4800 20A4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d2</td>
<td>0x4800 20A8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d3</td>
<td>0x4800 20AC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d4</td>
<td>0x4800 20B0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d5</td>
<td>0x4800 20B4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d6</td>
<td>0x4800 20B8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d7</td>
<td>0x4800 20BC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d8</td>
<td>0x4800 20C0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d9</td>
<td>0x4800 20C4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d10</td>
<td>0x4800 20C8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d11</td>
<td>0x4800 20CA</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d12</td>
<td>0x4800 20CB</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d13</td>
<td>0x4800 20D0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d14</td>
<td>0x4800 20D4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_d15</td>
<td>0x4800 20DA</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_ncs0</td>
<td>0x4800 2000</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_ncs1</td>
<td>0x4800 2080</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CLK[31:16]</td>
<td>gpmc_ncs2</td>
<td>0x4800 2080</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b111</td>
</tr>
<tr>
<td>REGISTER NAME</td>
<td>Pad Name</td>
<td>Physical Address</td>
<td>WakeUpx</td>
<td>OffMode</td>
<td>Input Enable</td>
<td>Reserved</td>
<td>PU/PD</td>
<td>MuxMode</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------</td>
<td>------------------</td>
<td>---------</td>
<td>---------</td>
<td>--------------</td>
<td>----------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_NCS3[15:0]</td>
<td>gpmc_ncs3</td>
<td>0x4800 20B4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_NCS3[31:16]</td>
<td>gpmc_ncs4</td>
<td>0x4800 20B4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_NCS5[15:0]</td>
<td>gpmc_ncs5</td>
<td>0x4800 20B8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_NCS5[31:16]</td>
<td>gpmc_ncs6</td>
<td>0x4800 20B8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_NCS7[15:0]</td>
<td>gpmc_ncs7</td>
<td>0x4800 20BC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_NCS7[31:16]</td>
<td>gpmc_clk</td>
<td>0x4800 20BC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_NADV_ALE[15:0]</td>
<td>gpmc_nadv_ale</td>
<td>0x4800 20C0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_NADV_ALE[31:16]</td>
<td>gpmc_noe</td>
<td>0x4800 20C0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_NWE[15:0]</td>
<td>gpmc_nwe</td>
<td>0x4800 20C4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_NWE[31:16]</td>
<td>gpmc_nbe0_cle</td>
<td>0x4800 20C4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_WAIT0[15:0]</td>
<td>gpmc_wait0</td>
<td>0x4800 20CC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_WAIT0[31:16]</td>
<td>gpmc_wait1</td>
<td>0x4800 20CC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_WAIT2[15:0]</td>
<td>gpmc_wait2</td>
<td>0x4800 20DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_GPMC_WAIT2[31:16]</td>
<td>gpmc_wait3</td>
<td>0x4800 20DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_PCLK[15:0]</td>
<td>dss_pclk</td>
<td>0x4800 20D4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_PCLK[31:16]</td>
<td>dss_hsync</td>
<td>0x4800 20D4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_VSYNC[15:0]</td>
<td>dss_vsync</td>
<td>0x4800 20DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_VSYNC[31:16]</td>
<td>dss_acbias</td>
<td>0x4800 20DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA0[15:0]</td>
<td>dss_data0</td>
<td>0x4800 20D4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA0[31:16]</td>
<td>dss_data1</td>
<td>0x4800 20D4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA2[15:0]</td>
<td>dss_data2</td>
<td>0x4800 20DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA2[31:16]</td>
<td>dss_data3</td>
<td>0x4800 20DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA4[15:0]</td>
<td>dss_data4</td>
<td>0x4800 20D4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA4[31:16]</td>
<td>dss_data5</td>
<td>0x4800 20D4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA6[15:0]</td>
<td>dss_data6</td>
<td>0x4800 20D8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA6[31:16]</td>
<td>dss_data7</td>
<td>0x4800 20D8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA8[15:0]</td>
<td>dss_data8</td>
<td>0x4800 20DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA8[31:16]</td>
<td>dss_data9</td>
<td>0x4800 20DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA10[15:0]</td>
<td>dss_data10</td>
<td>0x4800 20DF</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA10[31:16]</td>
<td>dss_data11</td>
<td>0x4800 20DF</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA12[15:0]</td>
<td>dss_data12</td>
<td>0x4800 20F4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
</tbody>
</table>

Table 7-76. CONTROL_PADCONF_CAPABILITIES (continued)
<table>
<thead>
<tr>
<th>REGISTER NAME</th>
<th>Pad Name</th>
<th>Physical Address</th>
<th>WakeUpx</th>
<th>OffMode</th>
<th>Input Enable</th>
<th>Reserved</th>
<th>PU/PD</th>
<th>MuxMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTROL_PADCONF_DSS_DATA12[31:16]</td>
<td>dss_data13</td>
<td>0x4800 20F4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA14[15:0]</td>
<td>dss_data14</td>
<td>0x4800 20F8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA14[31:16]</td>
<td>dss_data15</td>
<td>0x4800 20F8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA16[15:0]</td>
<td>dss_data16</td>
<td>0x4800 20FC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA16[31:16]</td>
<td>dss_data17</td>
<td>0x4800 20FC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA18[15:0]</td>
<td>dss_data18</td>
<td>0x4800 2100</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA18[31:16]</td>
<td>dss_data19</td>
<td>0x4800 2100</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA20[15:0]</td>
<td>dss_data20</td>
<td>0x4800 2104</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA20[31:16]</td>
<td>dss_data21</td>
<td>0x4800 2104</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA22[15:0]</td>
<td>dss_data22</td>
<td>0x4800 2108</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA22[31:16]</td>
<td>dss_data23</td>
<td>0x4800 2108</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA24[15:0]</td>
<td>dss_data24</td>
<td>0x4800 2112</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_DSS_DATA24[31:16]</td>
<td>dss_data25</td>
<td>0x4800 2112</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_HS[15:0]</td>
<td>cam_hs</td>
<td>0x4800 210C</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_HS[31:16]</td>
<td>cam_vs</td>
<td>0x4800 210C</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_XCLKA[15:0]</td>
<td>cam_xclka</td>
<td>0x4800 2110</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_XCLKA[31:16]</td>
<td>cam_pclk</td>
<td>0x4800 2110</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_FLD[15:0]</td>
<td>cam_fld</td>
<td>0x4800 2114</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_FLD[31:16]</td>
<td>cam_d0</td>
<td>0x4800 2114</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D1[15:0]</td>
<td>cam_d1</td>
<td>0x4800 2118</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D1[31:16]</td>
<td>cam_d2</td>
<td>0x4800 2118</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D3[15:0]</td>
<td>cam_d3</td>
<td>0x4800 211C</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D3[31:16]</td>
<td>cam_d4</td>
<td>0x4800 211C</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D5[15:0]</td>
<td>cam_d5</td>
<td>0x4800 2120</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D5[31:16]</td>
<td>cam_d6</td>
<td>0x4800 2120</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D7[15:0]</td>
<td>cam_d7</td>
<td>0x4800 2124</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D7[31:16]</td>
<td>cam_d8</td>
<td>0x4800 2124</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D9[15:0]</td>
<td>cam_d9</td>
<td>0x4800 2128</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D9[31:16]</td>
<td>cam_d10</td>
<td>0x4800 2128</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D11[15:0]</td>
<td>cam_d11</td>
<td>0x4800 212C</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D11[31:16]</td>
<td>cam_xclkb</td>
<td>0x4800 212C</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D13[15:0]</td>
<td>cam_wen</td>
<td>0x4800 2130</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CAM_D13[31:16]</td>
<td>cam_strobe</td>
<td>0x4800 2130</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CSI2_DX0[15:0]</td>
<td>csi2_dx0</td>
<td>0x4800 2134</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CSI2_DX0[31:16]</td>
<td>csi2_dy0</td>
<td>0x4800 2134</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b1</td>
<td>0b111</td>
</tr>
<tr>
<td>REGISTER NAME</td>
<td>Pad Name</td>
<td>Physical Address</td>
<td>WakeUpx</td>
<td>OffMode</td>
<td>Input Enable</td>
<td>Reserved</td>
<td>PU/PD</td>
<td>MuxMode</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>------------------</td>
<td>---------</td>
<td>---------</td>
<td>--------------</td>
<td>----------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CSI2_DX[15:0]</td>
<td>csi2_dx</td>
<td>0x4800 2138</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_CSI2_DX[31:16]</td>
<td>csi2_dy</td>
<td>0x4800 2138</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP2_FSX[15:0]</td>
<td>mcbsp2_fpx</td>
<td>0x4800 213C</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP2_FSX[31:16]</td>
<td>mcbsp2_clkx</td>
<td>0x4800 213C</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP2_DR[15:0]</td>
<td>mcbsp2_dr</td>
<td>0x4800 2140</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP2_DR[31:16]</td>
<td>mcbsp2_dx</td>
<td>0x4800 2140</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC1_CLK[15:0]</td>
<td>mmc1_clk</td>
<td>0x4800 2144</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC1_CLK[31:16]</td>
<td>mmc1_cmd</td>
<td>0x4800 2144</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC1_DAT0[15:0]</td>
<td>mmc1_dat0</td>
<td>0x4800 2148</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC1_DAT0[31:16]</td>
<td>mmc1_dat1</td>
<td>0x4800 2148</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC1_DAT2[15:0]</td>
<td>mmc1_dat2</td>
<td>0x4800 214C</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC1_DAT2[31:16]</td>
<td>mmc1_dat3</td>
<td>0x4800 214C</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_CLK[15:0]</td>
<td>mmc2_clk</td>
<td>0x4800 2150</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_CLK[31:16]</td>
<td>mmc2_cmd</td>
<td>0x4800 2150</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_DAT0[15:0]</td>
<td>mmc2_dat0</td>
<td>0x4800 215C</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_DAT0[31:16]</td>
<td>mmc2_dat1</td>
<td>0x4800 215C</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_DAT2[15:0]</td>
<td>mmc2_dat2</td>
<td>0x4800 2160</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_DAT2[31:16]</td>
<td>mmc2_dat3</td>
<td>0x4800 2160</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_DAT4[15:0]</td>
<td>mmc2_dat4</td>
<td>0x4800 2164</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_DAT4[31:16]</td>
<td>mmc2_dat5</td>
<td>0x4800 2164</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_DAT6[15:0]</td>
<td>mmc2_dat6</td>
<td>0x4800 2168</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_DAT6[31:16]</td>
<td>mmc2_dat7</td>
<td>0x4800 2168</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b11</td>
</tr>
</tbody>
</table>
SCM Register Manual

www.ti.com

Table 7-76. CONTROL_PADCONF_CAPABILITIES (continued)
REGISTER NAME

Pad Name

Physical
Address

WakeUpx

OffMode

Input
Enable

Reserved

PU/PD

MuxMode

222

CONTROL_PADCONF_UART2_TX[31:16]

uart2_rx

0x4800 2178

0b00

0b00000

0b1

0b000

0b11

0b111

CONTROL_PADCONF_UART1_TX[15:0]

uart1_tx

0x4800 217C

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_UART1_TX[31:16]

uart1_rts

0x4800 217C

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_UART1_CTS[15:0]

uart1_cts

0x4800 2180

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_UART1_CTS[31:16]

uart1_rx

0x4800 2180

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP4_CLKX[15:0]

mcbsp4_clkx

0x4800 2184

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP4_CLKX[31:16]

mcbsp4_dr

0x4800 2184

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP4_DX[15:0]

mcbsp4_dx

0x4800 2188

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP4_DX[31:16]

mcbsp4_fsx

0x4800 2188

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP1_CLKR[15:0]

mcbsp1_clkr

0x4800 218C

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP1_CLKR[31:16]

mcbsp1_fsr

0x4800 218C

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP1_DX[15:0]

mcbsp1_dx

0x4800 2190

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP1_DX[31:16]

mcbsp1_dr

0x4800 2190

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP_CLKS[15:0]

mcbsp_clks

0x4800 2194

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP_CLKS[31:16]

mcbsp1_fsx

0x4800 2194

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP1_CLKX[15:0]

mcbsp1_clkx

0x4800 2198

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_MCBSP1_CLKX[31:16]

uart3_cts_rctx

0x4800 2198

0b00

0b00000

0b1

0b000

0b11

0b111

CONTROL_PADCONF_UART3_RTS_SD[15:0]

uart3_rts_sd

0x4800 219C

0b00

0b00000

0b1

0b000

0b11

0b111

CONTROL_PADCONF_UART3_RTS_SD[31:16]

uart3_rx_irrx

0x4800 219C

0b00

0b00000

0b1

0b000

0b11

0b111

CONTROL_PADCONF_UART3_TX_IRTX[15:0]

uart3_tx_irtx

0x4800 21A0

0b00

0b00000

0b1

0b000

0b11

0b111

CONTROL_PADCONF_UART3_TX_IRTX[31:16]

hsusb0_clk

0x4800 21A0

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_STP[15:0]

hsusb0_stp

0x4800 21A4

0b00

0b00000

0b1

0b000

0b11

0b111

CONTROL_PADCONF_HSUSB0_STP[31:16]

hsusb0_dir

0x4800 21A4

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_NXT[15:0]

hsusb0_nxt

0x4800 21A8

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_NXT[31:16]

hsusb0_data0

0x4800 21A8

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_DATA1[15:0]

hsusb0_data1

0x4800 21AC

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_DATA1[31:16]

hsusb0_data2

0x4800 21AC

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_DATA3[15:0]

hsusb0_data3

0x4800 21B0

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_DATA3[31:16]

hsusb0_data4

0x4800 21B0

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_DATA5[15:0]

hsusb0_data5

0x4800 21B4

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_DATA5[31:16]

hsusb0_data6

0x4800 21B4

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_DATA7[15:0]

hsusb0_data7

0x4800 21B8

0b00

0b00000

0b1

0b000

0b01

0b111

CONTROL_PADCONF_HSUSB0_DATA7[31:16]

i2c1_scl

0x4800 21B8

0b00

0b00000

0b1

0b000

0b11

---

868 System Control Module

SPRUF98P – April 2010 – Revised March 2011
Submit Documentation Feedback
© 2010–2011, Texas Instruments Incorporated


<table>
<thead>
<tr>
<th>REGISTER NAME</th>
<th>Pad Name</th>
<th>Physical Address</th>
<th>WakeUpx</th>
<th>OffMode</th>
<th>Input Enable</th>
<th>Reserved</th>
<th>PU/PD</th>
<th>MuxMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTROL_PADCONF_I2C1_SDA[15:0]</td>
<td>i2c1_sda</td>
<td>0x4800 21BC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_I2C1_SDA[31:16]</td>
<td>i2c2_scl</td>
<td>0x4800 218C</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_I2C2_SDA[15:0]</td>
<td>i2c3_sda</td>
<td>0x4800 21C0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_I2C2_SDA[31:16]</td>
<td>i2c4_scl</td>
<td>0x4800 21C4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_I2C3_SDA[15:0]</td>
<td>hdq_sio</td>
<td>0x4800 21C4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI1_CLK[15:0]</td>
<td>mcspi1_clk</td>
<td>0x4800 21C8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI1_CLK[31:16]</td>
<td>mcspi1_simo</td>
<td>0x4800 21C8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI1_SOMI[15:0]</td>
<td>mcspi1_cs0</td>
<td>0x4800 21CC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI1_SOMI[31:16]</td>
<td>mcspi1_cs1</td>
<td>0x4800 21D0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI1_CS1[15:0]</td>
<td>mcspi1_cs2</td>
<td>0x4800 21D0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI1_CS1[31:16]</td>
<td>mcspi1_cs3</td>
<td>0x4800 21D4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI1_CS3[15:0]</td>
<td>mcspi2_clk</td>
<td>0x4800 21D4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI1_CS3[31:16]</td>
<td>mcspi2_simo</td>
<td>0x4800 21D8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI2_SIMO[15:0]</td>
<td>mcspi2_simo</td>
<td>0x4800 21D8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI2_SIMO[31:16]</td>
<td>mcspi2_cs0</td>
<td>0x4800 21DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCSPI2_SIMO[31:16]</td>
<td>mcspi2_cs1</td>
<td>0x4800 21DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SYS_NIRQ[15:0]</td>
<td>sys_nirq</td>
<td>0x4800 21E0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SYS_NIRQ[31:16]</td>
<td>sys_clkout2</td>
<td>0x4800 21E0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD0[15:0]</td>
<td>sad2d_mcad0</td>
<td>0x4800 21E4</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD0[31:16]</td>
<td>sad2d_mcad1</td>
<td>0x4800 21E4</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2[15:0]</td>
<td>sad2d_mcad2</td>
<td>0x4800 21E8</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2[31:16]</td>
<td>sad2d_mcad3</td>
<td>0x4800 21E8</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD4[15:0]</td>
<td>sad2d_mcad4</td>
<td>0x4800 21EC</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD4[31:16]</td>
<td>sad2d_mcad5</td>
<td>0x4800 21EC</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD6[15:0]</td>
<td>sad2d_mcad6</td>
<td>0x4800 21F0</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD6[31:16]</td>
<td>sad2d_mcad7</td>
<td>0x4800 21F0</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD8[15:0]</td>
<td>sad2d_mcad8</td>
<td>0x4800 21F4</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD8[31:16]</td>
<td>sad2d_mcad9</td>
<td>0x4800 21F4</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD10[15:0]</td>
<td>sad2d_mcad10</td>
<td>0x4800 21F8</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD10[31:16]</td>
<td>sad2d_mcad11</td>
<td>0x4800 21F8</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD12[15:0]</td>
<td>sad2d_mcad12</td>
<td>0x4800 21FC</td>
<td>--</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b000</td>
</tr>
<tr>
<td>REGISTER NAME</td>
<td>Pad Name</td>
<td>Physical Address</td>
<td>WakeUp</td>
<td>OffMode</td>
<td>Input Enable</td>
<td>Reserved</td>
<td>PU/PD</td>
<td>MuxMode</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>------------------</td>
<td>------------------</td>
<td>--------</td>
<td>---------</td>
<td>--------------</td>
<td>----------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD1(31:16)</td>
<td>sad2d_mcad10</td>
<td>0x4800 21FC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD1(15:0)</td>
<td>sad2d_mcad15</td>
<td>0x4800 2200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD1[31:16]</td>
<td>sad2d_mcad16</td>
<td>0x4800 2204</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD1(15:0)</td>
<td>sad2d_mcad17</td>
<td>0x4800 2204</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD1(31:16)</td>
<td>sad2d_mcad18</td>
<td>0x4800 2208</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD1(15:0)</td>
<td>sad2d_mcad19</td>
<td>0x4800 2208</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2(15:0)</td>
<td>sad2d_mcad20</td>
<td>0x4800 220C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2(31:16)</td>
<td>sad2d_mcad21</td>
<td>0x4800 220C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2(15:0)</td>
<td>sad2d_mcad22</td>
<td>0x4800 2210</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2[31:16]</td>
<td>sad2d_mcad23</td>
<td>0x4800 2210</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2(15:0)</td>
<td>sad2d_mcad24</td>
<td>0x4800 2214</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2[31:16]</td>
<td>sad2d_mcad25</td>
<td>0x4800 2214</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2(15:0)</td>
<td>sad2d_mcad26</td>
<td>0x4800 2218</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2[31:16]</td>
<td>sad2d_mcad27</td>
<td>0x4800 2218</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2(15:0)</td>
<td>sad2d_mcad28</td>
<td>0x4800 221C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD2[31:16]</td>
<td>sad2d_mcad29</td>
<td>0x4800 221C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD3(15:0)</td>
<td>sad2d_mcad30</td>
<td>0x4800 2220</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD3(31:16)</td>
<td>sad2d_mcad31</td>
<td>0x4800 2220</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD3(15:0)</td>
<td>sad2d_mcad32</td>
<td>0x4800 2224</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD3[31:16]</td>
<td>sad2d_mcad33</td>
<td>0x4800 2224</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD3(15:0)</td>
<td>sad2d_mcad34</td>
<td>0x4800 2228</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD3[31:16]</td>
<td>sad2d_mcad35</td>
<td>0x4800 2228</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD3(15:0)</td>
<td>sad2d_mcad36</td>
<td>0x4800 222C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MCAD3[31:16]</td>
<td>sad2d_clk26mi</td>
<td>0x4800 222C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_NRESPWRON(15:0)</td>
<td>sad2d_nrespwr0</td>
<td>0x4800 2230</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_NRESPWRON(31:16)</td>
<td>sad2d_nrespwelcome 0x4800 2230</td>
<td>0b00</td>
<td>0b000000</td>
<td>0b1</td>
<td>0b0000</td>
<td>0b01</td>
<td>0b0000</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_ARMNIRQ(15:0)</td>
<td>sad2d_arinniq</td>
<td>0x4800 2234</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b01</td>
<td>0b0000</td>
<td>0b01</td>
<td>0b0000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_ARMNIRQ(31:16)</td>
<td>sad2d_arinniq2</td>
<td>0x4800 2234</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b01</td>
<td>0b0000</td>
<td>0b01</td>
<td>0b0000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_SPINT(15:0)</td>
<td>sad2d_spint</td>
<td>0x4800 2238</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b01</td>
<td>0b0000</td>
<td>0b01</td>
<td>0b0000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_SPINT(31:16)</td>
<td>sad2d_spint2</td>
<td>0x4800 2238</td>
<td>0b00</td>
<td>0b0000</td>
<td>0b01</td>
<td>0b0000</td>
<td>0b01</td>
<td>0b0000</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_DMAREQ0(15:0)</td>
<td>sad2d_dmareq0</td>
<td>0x4800 223C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_DMAREQ0(31:16)</td>
<td>sad2d_dmareq1</td>
<td>0x4800 223C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 7-76. CONTROL_PADCONF_CAPABILITIES (continued)

<table>
<thead>
<tr>
<th>REGISTER NAME</th>
<th>Pad Name</th>
<th>Physical Address</th>
<th>WakeUp</th>
<th>OffMode</th>
<th>Input</th>
<th>Reserved</th>
<th>PU/PD</th>
<th>MuxMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTROL_PADCONF_SAD2D_DMAREQ2[15:0]</td>
<td>sad2d_dmareq2</td>
<td>0x4800 2240</td>
<td>---</td>
<td>0b--000</td>
<td>0b1</td>
<td>0b000</td>
<td>--</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_DMAREQ2[31:16]</td>
<td>sad2d_dmareq3</td>
<td>0x4800 2240</td>
<td>---</td>
<td>0b--000</td>
<td>0b1</td>
<td>0b000</td>
<td>--</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_NTRST[15:0]</td>
<td>sad2d_ntrst</td>
<td>0x4800 2244</td>
<td>---</td>
<td>0b--000</td>
<td>0b1</td>
<td>0b000</td>
<td>--</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_NTRST[31:16]</td>
<td>sad2d_tdi</td>
<td>0x4800 2244</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_TDO[15:0]</td>
<td>sad2d_tdo</td>
<td>0x4800 2248</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>--</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_TDO[31:16]</td>
<td>sad2d_tms</td>
<td>0x4800 2248</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>--</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_TCK[15:0]</td>
<td>sad2d_tck</td>
<td>0x4800 224C</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_TCK[31:16]</td>
<td>sad2d_tms</td>
<td>0x4800 224C</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MSTDBY[15:0]</td>
<td>sad2d_mstdby</td>
<td>0x4800 2250</td>
<td>0b00</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_MSTDBY[31:16]</td>
<td>sad2d_idlereq</td>
<td>0x4800 2250</td>
<td>--</td>
<td>0b--000</td>
<td>0b1</td>
<td>0b000</td>
<td>--</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_IDLEACK[15:0]</td>
<td>sad2d_idleack</td>
<td>0x4800 2254</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_IDLEACK[31:16]</td>
<td>sad2d_mwrite</td>
<td>0x4800 2254</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_SWRITE[15:0]</td>
<td>sad2d_swrite</td>
<td>0x4800 2258</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_SWRITE[31:16]</td>
<td>sad2d_mread</td>
<td>0x4800 2258</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_SREAD[15:0]</td>
<td>sad2d_sread</td>
<td>0x4800 225C</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_SREAD[31:16]</td>
<td>sad2d_mbusflag</td>
<td>0x4800 225C</td>
<td>--</td>
<td>0b00--0</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_SBUSFLAG[15:0]</td>
<td>sad2d_sbusflag</td>
<td>0x4800 2260</td>
<td>--</td>
<td>0b--000</td>
<td>0b1</td>
<td>0b000</td>
<td>--</td>
<td>---</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SAD2D_SBUSFLAG[31:16]</td>
<td>sdrc_cke0</td>
<td>0x4800 2260</td>
<td>--</td>
<td>---</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_SDRC_CKE1[15:0]</td>
<td>sdrc_cke1</td>
<td>0x4800 2264</td>
<td>--</td>
<td>---</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b111</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_CLK[15:0]</td>
<td>etk_clk</td>
<td>0x4800 25D8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_CLK[31:16]</td>
<td>etk_ctl</td>
<td>0x4800 25D8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D0[15:0]</td>
<td>etk_d0</td>
<td>0x4800 25DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D0[31:16]</td>
<td>etk_d1</td>
<td>0x4800 25DC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D2[15:0]</td>
<td>etk_d2</td>
<td>0x4800 25E0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D2[31:16]</td>
<td>etk_d3</td>
<td>0x4800 25E0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b11</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D4[15:0]</td>
<td>etk_d4</td>
<td>0x4800 25E4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D4[31:16]</td>
<td>etk_d5</td>
<td>0x4800 25E4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D6[15:0]</td>
<td>etk_d6</td>
<td>0x4800 25E8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D6[31:16]</td>
<td>etk_d7</td>
<td>0x4800 25E8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D8[15:0]</td>
<td>etk_d8</td>
<td>0x4800 25EC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D8[31:16]</td>
<td>etk_d9</td>
<td>0x4800 25EC</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D10[15:0]</td>
<td>etk_d10</td>
<td>0x4800 25F0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D10[31:16]</td>
<td>etk_d11</td>
<td>0x4800 25F0</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>REGISTER NAME</td>
<td>Pad Name</td>
<td>Physical Address</td>
<td>WakeUp</td>
<td>OffMode</td>
<td>Input Enable</td>
<td>Reserved</td>
<td>PU/PD</td>
<td>MuxMode</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------</td>
<td>------------------</td>
<td>--------</td>
<td>---------</td>
<td>--------------</td>
<td>----------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D12[15:0]</td>
<td>etk_d12</td>
<td>0x4800 25F4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D12[31:16]</td>
<td>etk_d13</td>
<td>0x4800 25F4</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D14[15:0]</td>
<td>etk_d14</td>
<td>0x4800 25F8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>CONTROL_PADCONF_ETK_D14[31:16]</td>
<td>etk_d15</td>
<td>0x4800 25F8</td>
<td>0b00</td>
<td>0b00000</td>
<td>0b1</td>
<td>0b000</td>
<td>0b01</td>
<td>0b100</td>
</tr>
<tr>
<td>Register Name</td>
<td>Physical Address</td>
<td>Mode 0</td>
<td>Mode 1</td>
<td>Mode 2</td>
<td>Mode 3</td>
<td>Mode 4</td>
<td>Mode 5</td>
<td>Mode 6</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_DATA[31:16]</td>
<td>0x4800 2164</td>
<td>mmc2_dat5</td>
<td>mmc2_dr_dat1</td>
<td>cam_global_rese</td>
<td>mmc3_dat1</td>
<td>gpio_137</td>
<td>hsusb3_lf_slp</td>
<td>mm3_nxdp</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MMC2_DATA[15:0]</td>
<td>0x4800 2166</td>
<td>mmc2_dat8</td>
<td>mmc2_dr_cmd</td>
<td>cam_shutter</td>
<td>mmc3_dat2</td>
<td>gpio_138</td>
<td>hsusb3_lf_dr</td>
<td>mm3_nxdp</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP2_DX[15:0]</td>
<td>0x4800 216C</td>
<td>mcbsp3_dx</td>
<td>uart2_clts</td>
<td>mcbsp3_clk_s</td>
<td>gpio_139</td>
<td>hsusb2_lf_data4</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP2_DX[31:16]</td>
<td>0x4800 216C</td>
<td>mcbsp3_dr</td>
<td>uart2_rs</td>
<td>mcbsp3_clk_s</td>
<td>gpio_140</td>
<td>hsusb2_lf_data5</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP2_CLKX[15:0]</td>
<td>0x4800 2170</td>
<td>mcbsp3_clkx</td>
<td>uart2_lx</td>
<td>mcbsp3_clk_s</td>
<td>gpio_142</td>
<td>hsusb2_lf_data6</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP2_CLKX[31:16]</td>
<td>0x4800 2170</td>
<td>mcbsp3_fsz</td>
<td>uart2_rx</td>
<td>mcbsp3_clk_s</td>
<td>gpio_143</td>
<td>hsusb2_lf_data7</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
</tr>
<tr>
<td>CONTROL_PADCONF_UART2_CTS[15:0]</td>
<td>0x4800 2174</td>
<td>uart2_cts</td>
<td>mcbsp3_dx</td>
<td>gpio_144</td>
<td>hsusb2_lf_data8</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_UART2_CTS[31:16]</td>
<td>0x4800 2174</td>
<td>uart2_rts</td>
<td>mcbsp3_dr</td>
<td>gpio_145</td>
<td>hsusb2_lf_data9</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_UART2_TX[15:0]</td>
<td>0x4800 2178</td>
<td>uart1_tx</td>
<td>mcbsp1_clk_r</td>
<td>gpio_146</td>
<td>hsusb2_lf_data10</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_UART2_TX[31:16]</td>
<td>0x4800 2178</td>
<td>uart1_rx</td>
<td>mcbsp1_clk_r</td>
<td>gpio_148</td>
<td>hsusb2_lf_data11</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_UART1_CTS[15:0]</td>
<td>0x4800 217C</td>
<td>uart1_cts</td>
<td>mcbsp1_clk_r</td>
<td>gpio_149</td>
<td>hsusb2_lf_data12</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_UART1_CTS[31:16]</td>
<td>0x4800 217C</td>
<td>uart1_rs</td>
<td>mcbsp1_clk_r</td>
<td>gpio_150</td>
<td>hsusb2_lf_data13</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_UART1_TX[15:0]</td>
<td>0x4800 2180</td>
<td>uart1_cts</td>
<td>mcbsp1_clk_r</td>
<td>gpio_151</td>
<td>hsusb2_lf_data14</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_UART1_TX[31:16]</td>
<td>0x4800 2180</td>
<td>uart1_rx</td>
<td>mcbsp1_clk_r</td>
<td>gpio_152</td>
<td>hsusb2_lf_data15</td>
<td>mm3_nxdp</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKR[15:0]</td>
<td>0x4800 2184</td>
<td>mcbsp1_clk_r</td>
<td>mcbsp1_clk_r</td>
<td>gpio_153</td>
<td>hsusb3_lf_data1</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKR[31:16]</td>
<td>0x4800 2184</td>
<td>mcbsp1_fsr</td>
<td>adpllv2d_dither</td>
<td>gpio_155</td>
<td>hsusb3_lf_data2</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKX[15:0]</td>
<td>0x4800 2188</td>
<td>mcbsp1_clkx</td>
<td>mcbsp1_clkx</td>
<td>gpio_156</td>
<td>hsusb3_lf_data3</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKX[31:16]</td>
<td>0x4800 2188</td>
<td>mcbsp1_fsx</td>
<td>mcbsp1_fsx</td>
<td>gpio_158</td>
<td>hsusb3_lf_data4</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP_CLKS[15:0]</td>
<td>0x4800 2194</td>
<td>mcbsp1_clkx</td>
<td>mcbsp1_clkx</td>
<td>gpio_159</td>
<td>hsusb3_lf_data5</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP_CLKS[31:16]</td>
<td>0x4800 2194</td>
<td>mcbsp1_fsx</td>
<td>mcbsp1_fsx</td>
<td>gpio_160</td>
<td>hsusb3_lf_data6</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKR[15:0]</td>
<td>0x4800 2198</td>
<td>mcbsp1_clk_r</td>
<td>mcbsp1_clk_r</td>
<td>gpio_161</td>
<td>hsusb3_lf_data7</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKR[31:16]</td>
<td>0x4800 2198</td>
<td>mcbsp1_fsr</td>
<td>adpllv2d_dither</td>
<td>gpio_163</td>
<td>hsusb3_lf_data8</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKX[15:0]</td>
<td>0x4800 219C</td>
<td>mcbsp1_clk_x</td>
<td>mcbsp1_clk_x</td>
<td>gpio_164</td>
<td>hsusb3_lf_data9</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKX[31:16]</td>
<td>0x4800 219C</td>
<td>mcbsp1_fsz</td>
<td>adpllv2d_dither</td>
<td>gpio_165</td>
<td>hsusb3_lf_data10</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP_CLKS[15:0]</td>
<td>0x4800 2198</td>
<td>mcbsp1_clkx</td>
<td>mcbsp1_clkx</td>
<td>gpio_166</td>
<td>hsusb3_lf_data11</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP_CLKS[31:16]</td>
<td>0x4800 2198</td>
<td>mcbsp1_fsx</td>
<td>mcbsp1_fsx</td>
<td>gpio_167</td>
<td>hsusb3_lf_data12</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKR[15:0]</td>
<td>0x4800 2198</td>
<td>mcbsp1_clk_r</td>
<td>mcbsp1_clk_r</td>
<td>gpio_168</td>
<td>hsusb3_lf_data13</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKR[31:16]</td>
<td>0x4800 2198</td>
<td>mcbsp1_fsr</td>
<td>adpllv2d_dither</td>
<td>gpio_169</td>
<td>hsusb3_lf_data14</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKX[15:0]</td>
<td>0x4800 2198</td>
<td>mcbsp1_clk_x</td>
<td>mcbsp1_clk_x</td>
<td>gpio_170</td>
<td>hsusb3_lf_data15</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP1_CLKX[31:16]</td>
<td>0x4800 2198</td>
<td>mcbsp1_fsz</td>
<td>adpllv2d_dither</td>
<td>gpio_172</td>
<td>hsusb3_lf_data16</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP_CLKS[15:0]</td>
<td>0x4800 2198</td>
<td>mcbsp1_clkx</td>
<td>mcbsp1_clkx</td>
<td>gpio_173</td>
<td>hsusb3_lf_data17</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
<tr>
<td>CONTROL_PADCONF_MCBSP_CLKS[31:16]</td>
<td>0x4800 2198</td>
<td>mcbsp1_fsx</td>
<td>mcbsp1_fsx</td>
<td>gpio_174</td>
<td>hsusb3_lf_data18</td>
<td>mm3 lsb</td>
<td>safe_mode</td>
<td></td>
</tr>
</tbody>
</table>