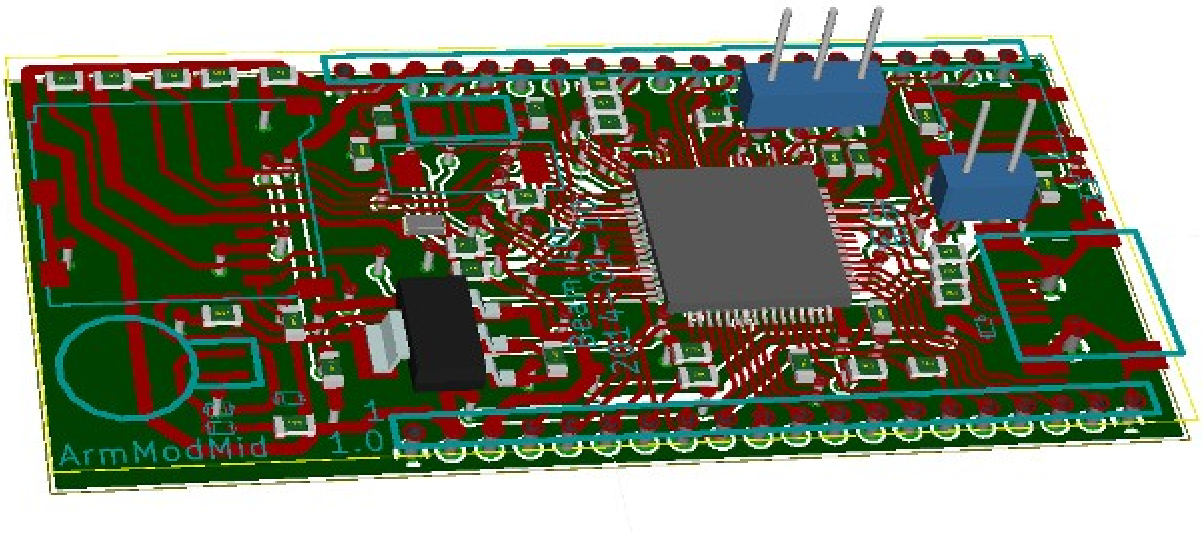
ArmModMid

ARM 32bit processor Module - Beam Ltd

Version 1.0 – 2014-02-02

*** Preliminary ***



1 What Is It ?

For hobbyist and educational electronics the usage of small 8bit micro-processors in simple and relatively large, dual in-line packages, have been the norm. The Microchip PIC series and Atmel AVR series are very popular and PCB's can be easily produced ans assembled with them. More recently there has been a big push in low cost and low power 32bit ARM based microprocessors. These are a lot more powerful than 8bit ones, and due to their 32bit architecture are easier to program. Unfortunately they only tend to come in small packages with fine pin spacing that make it difficult to use for the hobbyist and student when making their own electronics PCB designs.

The idea of the ArmModMid unit is to provide a simple and small ARM processor based module that can be installed onto hobbyist and student designed and built electronics PCB's allowing them to easily make use of this technology. It connects to the users PCB through two 20way 2mm pitch connectors or soldered pinholes. The module is relatively small at 59mm x 29mm.

As well as a 32bit ARM STM32M4 processor, it has a USB interface, 14 ADC channels, a MicroSD card holder and a battery backed up real-time clock amongst other features making it especially useful as a data logger, robot control and general control module.

2 Features

The ArmModMid is designed with data logging usage in mind. It has the following core features:

- Powerful ARM 32bit processor running at up to 168MHz with floating point hardware. STM32F405.
- Processor has 1 MByte of Flash program storage and 192kbytes of RAM
- Mini USB-B interface for power, programming, debug and general usage (communications with a PC for example).
- MicroSD card socket allowing up to 64GBytes of data storage memory to be used.
- Battery backed up real-time clock to store and maintain date and time.

- 5V to 3.3V regulator for local and optionally external power.
- 40pin 2mm pitch connector to main PCB.
- Debug LED. Every microprocessor needs one of these :)

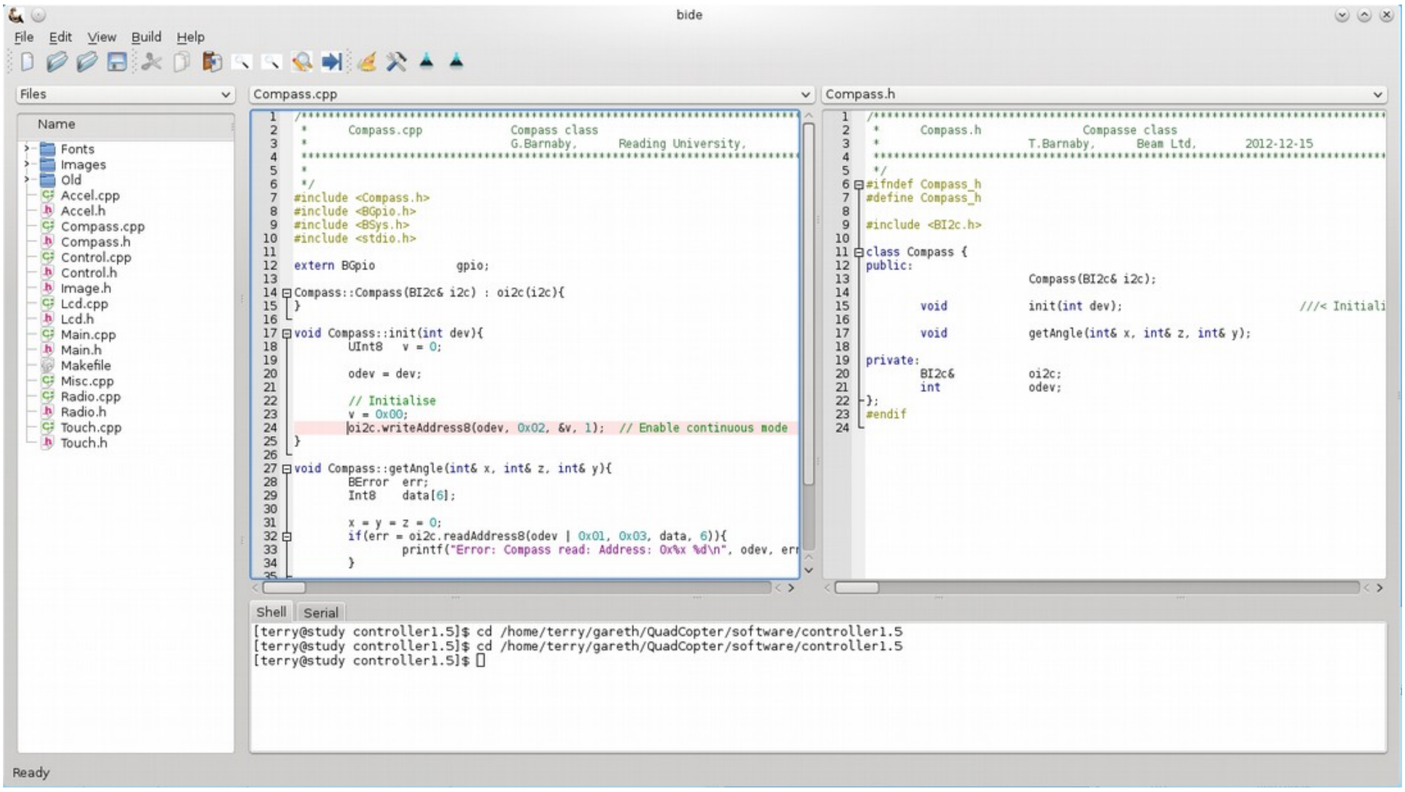
The ARM STM32F405 on the ArmModMid provides the following core features:

CPU	32bit ARM core running at up to 168MHz
FLASH	1 Mbyte
RAM	192 kbytes
STORE	4 kbytes of battery backed up RAM for non-volatile memory store
USB	USB 2.0 high-speed/full-speed interface
ADC	14 channels into 3 x 12bit ADC's at up to 2.4 mega samples per second
DAC	2 channels from 2 x 12bit DAC's
TIMERS	14 timers (16 and 32bit)
PWM	Up to 16 PWM channels
I2C	2 x I2C interfaces
SPI	2 x SPI interfaces
SERIAL	4 x serial port UARTS
CAN	2 x CAN bus
RTC	Battery backed up real time clock for date and time
DIGITAL IO	31 lines of general digital IO lines shared with peripherals

3 Software Overview

The 32bit ARM core provides a lot of power to create sophisticated systems. To help with harnessing this power in an easy to use way we provide the ArmSys software system. This provides a simple to use integrated development environment with GNU C++/C compiller tool set and the ArmSys C++ class library to make programing the systems relatively easy along with examples.
Features of the ArmSys environment are:

- Simple C++ class library to access system hardware including I2C, SPI, RTC, ADC's etc.
- Can be used as a simple single program or as a multi tasking real time program.
- Class library provides easy usage of the real time operating system features of FreeRTOS allowing multiple tasks to be run simultaneously,
- Class library provides useful modules such as a MicroSD card file system and USB disk device support.



```
/*******************************************************************************
 *      Compass.cpp             Compass class
 *                              G.Barnaby,      Reading University,
 *******************************************************************************
 *
 */
#include <Compass.h>
#include <BGpio.h>
#include <BSys.h>
#include <stdio.h>

extern BGpio            gpio;

Compass::Compass(BI2c& i2c) : oi2c(i2c){
}

void Compass::init(int dev){
        UInt8   v = 0;

        odev = dev;

        // Initialise
        v = 0x00;
        oi2c.writeAddress8(odev, 0x02, &v, 1);  // Enable continuous mode
}

void Compass::getAngle(int& x, int& z, int& y){
        BError  err;
        Int8    data[6];

        x = y = z = 0;
        if(err = oi2c.readAddress8(odev | 0x01, 0x03, data, 6)){
                printf("Error: Compass read: Address: 0x%x %d\n", odev, err
        }
```

```
/*******************************************************************************
 *      Compass.h                       Compasse class
 *                              T.Barnaby,      Beam Ltd,       2012-12-15
 *******************************************************************************
 */
#ifndef Compass_h
#define Compass_h

#include <BI2c.h>

class Compass {
public:
                                Compass(BI2c& i2c);

        void                    init(int dev);                          ///< Initiali

        void                    getAngle(int& x, int& z, int& y);

private:
        BI2c&                   oi2c;
        int                     odev;
};
#endif
```

```
[terry@study controller1.5]$ cd /home/terry/gareth/QuadCopter/software/controller1.5
[terry@study controller1.5]$ cd /home/terry/gareth/QuadCopter/software/controller1.5
[terry@study controller1.5]$ 
```

4 Hardware

The ArmModMid module comes ready to use and can be plugged into the USB port of a PC using a USB cable with a Mini-B sytle connector. To design a PCB using it we provide a Kicad schematic symbol and PCB footprint library files. This makes it easy to produce your PCB design.

It is a 3.3V based module so all input/output pins should be between 0V and 3.3V. In fact most of the IO pins are 5V tolerant. The only ones that are not are the PA4 and PA5 lines.

4.1 Powering

The ArmModMid can be powered through its USB connection. This has a diode from the USB 5V input to the boards +5V pins. However for most stand-alone operations it will require a power supply. It has an on-board 3.3V regulator to power the system which will work from an input voltage between 4.7V and 14V provided on its +5V input pins. So for a battery supply 6V would be ideal.

It is also possible to provide the 3.3V supply to the board from an external regulator if so desired allowing operation from lower voltage batteries or other sources.

Source	Pin	Description
USB 5V	USB connector	5V from the USB connector through a diode to the +5V board pins and on board 3.3V regulator
4.7 to 14V	+5V pins	External 4.7 to 14V input voltage. On board 3.3V regulator provides 3.3V for operation.
3.3V	+3.3V pins	External 3.3V input voltage.

Power consumption will depend on clock frequency used and the number of hardware peripherals enabled. It is likely to be in the range from 2mA to 95mA at 3.3V. A typical power usage would be around 40mA at 3.3V.

5 Software

The CPU is a standard ST Micro STM32M405 32bit ARM chip. There are many software development systems available for this architecture taht can be used. We will describe our own system, ArmSys. This is a development environment based on the GNU C++/C compiler with a GUI IDE, a class library and optional real-time kernel environment.

*** TBD ***

5.1 Programming The Device

The unit is easily programmed through the USB interface. There is one jumper on the board, J1, that enables programming. A jumper should be installed here and the module reset using the reset button or just powered up. Once done the development software on the PC will see the device and will be able to program it. Once program development is completed the jumper can be removed and the board will automatically run the last programmed software.

6 Connector Pinouts

The tables shows the ArmModMid connector pinouts with the main functions of each pin. Please look at the STM32M405 datasheets for more uses of each pin.

Pin	Usage	Notes
P1.1	GND	Power ground
P1.2	PC6/SER6_TX/T8.1/T3.1	
P1.3	PC7/SER6_RX/T8.2/T3.2	
P1.4	PA10/T1.3	PWM-T1.3
P1.5	PA15/SPI3_CS0/T2.1-ETR	
P1.6	PB3/SPI3_CLK/T2.2	
P1.7	PB4/SPI3_MISO/T3.1	
P1.8	PB5/SPI3_MOSI/T3.2	
P1.9	PB6/SER1_TX/T4.1	
P1.10	PB7/SER1_RX/T4.2	
P1.11	PB8/I2C1_CLK/T4.3/T10.1/CAN1_RX	
P1.12	PB9/I2C1_DAT/T4.4/T11.1/CAN1_TX	
P1.13	GND	Power ground (near analogue inputs)
P1.14	PC0/A10	
P1.15	PC1/A11	
P1.16	PC2/A12	
P1.17	PC3/A13	
P1.18	+5V	5Volts in or out from USB
P1.19	+3.3V	3.3V out from regulator
P1.20	GND	Power ground
P2.1	GND	Power ground
P2.2	+3.3V	3.3V out from regulator
P2.3	+5V	5Volts in or out from USB
P2.4	PA0/SER4_TX/A0/T5.1/T2.1-ETR	

Pin	Usage	Notes
P2.5	PA1/SER4_RX/A1/T5.2/T2.2	
P2.6	PA2/SER2_TX/A2/T5.3/T9.1/T2.3	
P2.7	PA3/SER2_RX/A3/T5.4/T9.2/T2.4	
P2.8	PA4/SPI1_CS0/A4/	
P2.9	PA5/SPI1_CLK/A5	
P2.10	PA6/SPI1_MISO/A6/T13.1/T3.1	
P2.11	PA7/SPI1_MOSI/A7T14.1/T3.2	
P2.12	PB0/A8/T3.3	
P2.13	PB1/A9/T3.4	
P2.14	PB10/I2C2_CLK/SER3_TX/T2.3	
P2.15	PB11/I2C2_DAT/SER3_RX/T2.4	
P2.16	PB12/CAN2_RX	
P2.17	PB13/CAN2_TX	
P2.18	PB14/T12.1/U2M	Timer12.1 and second USB-
P2.19	PB15/T12.2/U2P	Timer12.2 and second USB+
P2.20	GND	Power ground
	PC13/LED	On board LED for debug

7 Further Information

The ArmModMid user manual	ArmModMidManual.pdf
STM32 overal description	doc/STM32-overview.pdf
STM32M405 electrical manual	doc/STM32F405xx-electrical.pdf
STM32M405 reference manual	doc/STM32F405xx-reference.pdf
STM32M405 programming manual	doc/STM32F405xx-programming.pdf