Table of Contents

1 Introduction

1.1 Bambino Family of Controllers

The Micromint Bambino 210 is a multi-core SBC designed for compatibility with the mbed framework. This popular framework for embedded software development includes extensive class libraries, tools and broad community support, allowing you to deliver embedded applications faster and more reliably. The Micromint Bambino 210 is powered by an NXP LPC4330, the first dual-core ARM Cortex-M microcontroller. Its Cortex-M4 and Cortex-M0 cores are both capable of concurrent 204 MHz operation. The hardware block diagram and feature summary is included below:

1.2 Bambino 210 Base Features

- 204 MHz Dual Core 32-bit ARM® Cortex?-M4/M0
- 264k SRAM
- 4M SPIFI Flash
- USB Device Port
- 2 Push Buttons
- 4 LEDs
- Arduino-compatible headers

1.3 Bambino 210E Features

- Same Base Features
- 8M SPIFI Flash
- 10 Gadgeteer Sockets
- 100 Mbps Ethernet
- microSD Socket
- Xbee Socket

1.4 Software and Support

NEXT: Getting Started

PREVIOUS: Table of Contents

2 Getting Started

2.1 Getting Started

This section explains how to develop programs for the Micromint Bambino using the mbed online compiler. Using an online compiler is a great way to get started with mbed development. No local program installations are required on your system. The editor, compiler and linker are maintained by the mbed team and accessible via any Internet connection. The next chapter will cover offline toolchains for projects that require use of a compiler and debugger installed on your system.

2.1.1 Activate mbed.org Account

To use the online compiler you first need a developer account at mbed.org. If you have not done so, please visit mbed.org, click on "Signup" and follow the steps to create an account. If you already have an account, login using your mbed username.

2.1.2 Add Micromint Bambino Platforms

The Micromint Bambino has two USB ports: USB0 for use by your mbed applications and USB1 for mbed development. When you connect your system to USB1 it will recognize the board as a removable flash drive to copy your programs and a serial port that can be used for program output. Those functions are implemented by the on-board mbed HDK with the mbed CMSIS-DAP firmware. To add the board to the online compiler, click on the mbed.htm file in the removable flash drive detected by your system.

Equivalent links are included below.

Add Micromint Bambino 210E

Add Micromint Bambino 210

2.1.3 Create Program

Enter the online compiler by selecting the "Compiler" button on the development site. Right click on "My Programs" and select "New Program...".

Select a name for your program. Use "blinky" as our initial program name for a simple application that blinks an LED.

Then create your first program file. Right click on "blinky" and select "New File...". Use "main.cpp" as your main program file.

Paste the following content to your program file.

```
#include "mbed.h"

DigitalOut myled(LED2);

int main() {
    while(1) {
        myled = 1;
        wait(0.8);
        myled = 0;
        wait(0.8);
    }
}
```

Finally import the mbed source library to your program to access mbed library functions with source code. Right click on "blinky" and select "Import Library...", "From Import Wizard". On the wizard search for "mbed-src" and double click on the latest version from "mbed-official". It will take a few seconds to import all library files to your program.

2.1.4 Compile and Download Program

To compile your program click "Compile" on the toolbar. This will create a program binary to run on your board and save it to your system. You can save the file directly to the board by selecting the mbed removable flash drive as the target. If you prefer to keep a copy on your system, save it first to your hard drive and then copy it to the mbed removable flash drive. If you get any compiler errors, check the source code to insure it is as listed above and then recompile your program.

2.1.5 Run Program

Once you copy your program to the board, the green LED (LED2) should start blinking. If your previous program left the board in an unknown state, press the RESET button to load the program you just copied.

That's it! In less than 5 minutes you were able to compile and run a program on your board using the online compiler. The next chapter will cover how to use offline compilers.

NEXT: Offline Toolchains

PREVIOUS: Introduction

3 Offline Toolchains

3.1 Offline Toolchains

The online compiler covered in the previous chapter is the fastest way to start programming mbed applications. There is no software to install and the toolchain is actively managed for you. Yet some projects require full debugging capabilities, custom library builds, group source code control and other options not currently available on the online compiler. This chapter covers how to use offline toolchains to build both your application as well as the mbed library.

The Micromint Bambino implements the mbed HDK on-board using an additional microcontroller (NXP LPC11U35) with the mbed CMSIS-DAP firmware. Offline compilers can use the mbed HDK for debugging by configuring their toolchain debug options to CMSIS-DAP. To use external JTAG probes an alternate firmware is required as described in Appendix B.

3.2 Keil MDK

Keil MDK is a popular toolchain for development of ARM applications in C or C++. The free MDK-Lite allows you to build applications up to 32KB. Larger binaries require a commercial license. Many mbed examples include Makefiles for Keil MDK, including the mbed Book Examples for the Micromint Bambino. This section explains how to build the blinky mbed book example using Keil.

Select "Project", "Open Project..." and change to the mbed-book-02-06\PE_02-01_SimpleLED directory. Select the project file PE_02-01_SimpleLED.uvproj. Your IDE should look similar to the one below:

To use the on-board debugger please use the configuration listed in Appendix C. To build a binary from the IDE just press F7 or click on the compile toolbar button. To download the binary and start a debug session press Ctrl-F5 or click on the debug button. Your debug environment should look as follows:

Press F5 or clock the run button and the yellow LED (LED1) should start blinking. Press the stop button and place a breakpoint on the first line of the while loop by clicking to the left side of the line. Then press F5 to see the LED blink once. You can examine registers and perform other debugging options.

Developers creating new device drivers or changing mbed library features may find it convenient to include the library sources in their project files. The blinky_src project is a simple example to get started. The project can include your application source code as well as full mbed library sources.

3.3 GCC-ARM

GCC-ARM is a freely available GNU toolchain for ARM Cortex microcontrollers. It is actively maintained, regression tested and includes binaries for Windows, Linux and Mac. Many mbed examples include Makefiles for GCC-ARM, including the mbed Book Examples for the Micromint Bambino. This section explains how to build the blinky mbed book example using GCC-ARM.

After installing GCC-ARM verify that the ARM 'gcc' compiler and 'make' can be found on your PATH.

```
C:\Users\mbed>arm-none-eabi-gcc -v
Using built-in specs.
COLLECT_GCC=arm-none-eabi-gcc
COLLECT_LTO_WRAPPER= ...
Target: arm-none-eabi
Configured with: /home/build/work/GCC-4-8-build/src/gcc/configure ...
Thread model: single
gcc version 4.8.4 20140526 (release) [ARM/embedded-4_8-branch revision 211358]
  (GNU Tools for ARM Embedded Processors)

C:\Users\mbed>make -v
GNU Make 3.82.90
Built for i686-pc-mingw32
Copyright (C) 1988-2012 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
```

Note that GCC-ARM does not include 'make'. Developers using Windows can get binaries for 'make' and other utilities from unxutils or other projects.

Change to the project directory, build the binary and copy it to the mbed removable drive.

```
C:\Users\mbed>cd mbed-book-02-06\PE_02-01_SimpleLED

C:\Users\mbed\mbed-book-02-06\PE_02-01_SimpleLED>make
arm-none-eabi-g++ ... main.cpp
arm-none-eabi-gcc ... -lc -lgcc -lnosys
arm-none-eabi-objcopy -O binary PE_02-01_SimpleLED.elf PE_02-01_SimpleLED.bin

C:\Users\mbed\mbed-book-02-06\PE_02-01_SimpleLED>copy PE_02-01_SimpleLED.bin E:
```

The yellow LED (LED1) should start blinking. If your previous program left the board in an unknown state, press the RESET button to load the program you just copied.

GCC-ARM includes a GDB debugger binary for ARM (arm-none-eabi-gdb) which can be used with the on-board mbed CMSIS-DAP debugger via pyOCD or OpenOCD (0.8.0 or above).

3.4 Rebuilding mbed Libraries

The source code for the mbed library is open source and can be downloaded from the mbed mainline repository or the Micromint repository. To recompile it please follow these steps:

1. Install a compatible ARM toolchain. The mbed port for LPC4330 targets currently supports the Keil MDK, GCC ARM and GCC LPCXpresso toolchains. The Keil MDK is the main ARM toolchain at Micromint.

2. Install Python 2.7. The mbed build scripts are written in Python.

3. Install the mbed Library Sources. This includes full source code for all supported multiple microcontrollers and boards.

4. Edit workspace_tools/private_settings.py to reflect your toolchain directories.

```
from os.path import join

# ARM
armcc = "keil"
ARM_PATH = "C:/Keil/ARM"
ARM_BIN = join(ARM_PATH, "ARMCC", "bin")
ARM_INC = join(ARM_PATH, "ARMCC", "include")
ARM_LIB = join(ARM_PATH, "ARMCC", "lib")

# GCC ARM
GCC_ARM_PATH = "C:/Program Files (x86)/GNU Tools ARM Embedded/4.8 2014q2/bin"

# GCC CodeRed
GCC_CR_PATH = "C:/nxp/LPCXpresso_7.5.0_254/lpcxpresso/tools/bin"
```

5. Compile the base mbed libraries. These are typical commands to change to the source directory, setup compiler environment and build the bootloader. Note that the Python directory in the PATH may be different in your development system.

```
cd mbed-master
SET PATH=C:\Windows\system32;C:\Windows;C:\Python27
python workspace_tools/build.py -v -r -m LPC4330_M4 -t ARM > build.log
python workspace_tools/build.py -v -r -m LPC4330_M4 -t GCC_ARM >> build.log
```

If your build is successful, the mbed libraries and headers for the LPC4330 will be available in the ..\build directory. If not, check the build.log file. For more documentation on these procedures please visit the mbed tools site.

NEXT: Hardware

PREVIOUS: Getting Started

4 Hardware

The following image shows where some of the hardware components are located.

Bambino 210 Hardware

4.1 Microcontroller

The Bambino 210 includes a NXP LPC4330 microcontroller. These dual core 32-bit ARM Cortex-M4/M0 RISC microcontroller are capable of 204-MHz operation with a Thumb2 instruction set for smaller object code. It uses a Harvard architecture with separate local instruction and data buses as well as a separate peripherals bus. Please see NXP?s LPC4330 Microcontroller's User Manual for more information and register definitions.

4.1.1 LPC4330 key features

- Cortex-M4 Processor core
  - Built-in Memory Protection Unit (MPU) supporting eight regions.
  - Running at frequencies of up to 204 MHz.
  - Built-in Nested Vectored Interrupt Controller (NVIC).
  - Hardware floating-point unit.
  - Non-maskable Interrupt (NMI) input.
  - JTAG and Serial Wire Debug (SWD), serial trace, eight breakpoints, and four watch points.
  - Enhanced Trace Module (ETM) and Enhanced Trace Buffer (ETB) support.
  - System tick timer.
- Cortex-M0 Processor core
  - Running at frequencies of up to 204 MHz.
  - JTAG and built-in NVIC.
- On-chip memory
  - 264 kB SRAM for code and data use.
  - Multiple SRAM blocks with separate bus access. Two SRAM blocks can be powered down individually.
  - 64 kB ROM containing boot code and on-chip software drivers.
  - 128 bit general-purpose One-Time Programmable (OTP) memory.
- Clock generation unit
  - Crystal oscillator with an operating range of 1 MHz to 25 MHz.

- 12 MHz Internal RC (IRC) oscillator trimmed to 1 % accuracy over temperature and voltage.
- Ultra-low power Real-Time Clock (RTC) crystal oscillator.
- Three PLLs allow CPU operation up to the maximum CPU rate without the need for a high-frequency crystal. The second PLL is dedicated to the High-speed USB, the third PLL can be used as audio PLL.
- Clock output.

- Configurable digital peripherals
  - Serial GPIO (SGPIO) interface.
  - State Configurable Timer (SCT) subsystem on AHB.
  - Global Input Multiplexer Array (GIMA) allows to cross-connect multiple inputs and outputs to event driven peripherals like the timers, SCT, and ADC0/1.
- Serial interfaces
  - Quad SPI Flash Interface (SPIFI) with 1-, 2-, or 4-bit data at rates of up to 52 MB per second.
  - 10/100T Ethernet MAC with RMII and MII interfaces and DMA support for high throughput at low CPU load. Support for IEEE 1588 time stamping/advanced time stamping (IEEE 1588-2008 v2).
  - One High-speed USB 2.0 Host/Device/OTG interface with DMA support and on-chip high-speed PHY (USB0).
  - One High-speed USB 2.0 Host/Device interface with DMA support, on-chip full-speed PHY and ULPI interface to external high-speed PHY (USB1).
  - USB interface electrical test software included in ROM USB stack.
  - Four 550 UARTs with DMA support: one UART with full modem interface; one UART with IrDA interface; three USARTs support UART synchronous mode and a smart card interface conforming to ISO7816 specification.
  - Up to two C_CAN 2.0B controllers with one channel each.
  - Two SSP controllers with DMA, FIFO and multi-protocol support.
  - One SPI controller.
  - One Fast-mode Plus I2C-bus interface with monitor mode and with open-drain I/O pins conforming to the full I2C-bus specification. Supports data rates of up to 1 Mbit/s.
  - One standard I2C-bus interface with monitor mode and with standard I/O pins.
  - Two I2S interfaces, each with DMA support and with one input and one output.
- Digital peripherals
  - External Memory Controller (EMC) supporting external SRAM, ROM, NOR flash, and SDRAM devices.
  - Secure Digital Input Output (SD/MMC) card interface.
  - Eight-channel General-Purpose DMA controller can access all memories on the AHB and all DMA-capable AHB slaves.
  - General-Purpose Input/Output (GPIO) pins with configurable pull-up/pull-down resistors.
  - GPIO registers are located on the AHB for fast access. GPIO ports have DMA support.
  - Up to eight GPIO pins can be selected from all GPIO pins as edge and level sensitive interrupt sources.
  - Two GPIO group interrupt modules enable an interrupt based on a programmable pattern of input states of a group of GPIO pins.
  - Four general-purpose timer/counters with capture and match capabilities.
  - One motor control Pulse Width Modulator (PWM) for three-phase motor control.
  - One Quadrature Encoder Interface (QEI).
  - Repetitive Interrupt timer (RI timer).
  - Windowed watchdog timer (WWDT).
  - Ultra-low power Real-Time Clock (RTC) on separate power domain with 256 bytes of battery powered backup registers.
  - Alarm timer; can be battery powered.
- Analog peripherals
  - One 10-bit DAC with DMA support and a data conversion rate of 400 kSamples/s.
  - Two 10-bit ADCs with DMA support and a data conversion rate of 400 kSamples/s. Up to eight input channels per ADC.
- Unique ID for each device.
- Power
  - Single 3.3 V (2.2 V to 3.6 V) power supply with on-chip internal voltage regulator for the core supply and the RTC power domain.
  - RTC power domain can be powered separately by a 3 V battery supply.
  - Four reduced power modes: Sleep, Deep-sleep, Power-down, and Deep power-down.
  - Processor wake-up from Sleep mode via wake-up interrupts from various peripherals.
  - Wake-up from Deep-sleep, Power-down, and Deep power-down modes via external interrupts and interrupts generated by battery powered blocks in the RTC power domain.
  - Brownout detect with four separate thresholds for interrupt and forced reset.
  - Power-On Reset (POR).

4.1.2 LPC4330 Block Diagram

LPC4330 Block Diagram

4.1.3 LPC4330 Memory Map

LPC4330 Memory Map

4.2 Serial Flash Memory

The Bambino 210 uses Quad SPI Flash for it's program and non-volatile data storage. The quad SPI flash has a maximum clock rate of 80 MHz. The Bambino 210 uses a 4M flash and the Bambino 210E uses an 8M flash. Both memories have 4KB sectors.

4.3 USB Power Mux

Texas Instruments TPS2115 auto-switching power mux is used to select between USB0 and USB1. It provides a seamless transition between USB1 (MBED HDK) and USB0 on the Bambino 210. The TPS2115 includes thermal protection and reverse-conduction blocking.

4.4 Ethernet PHY (Bambino 210E)

The Bambino 210E includes a Micrel KSZ8081 10 Base-T/100 Base-TX Physical Layer Transceiver (PHY). The PHY has a RMII interface to transmit and receive data to the LPC4330's Media Access Controller. It has auto-negotiation to automatically select the highest link-up speed (10/100 Mbps) and duplex (half/full). For further information please see KSZ8081 Data Sheet.

4.5 MBED HDK

The MBED HDK is powered by NXP's LPC11U35. The mbed HDK uses the CMSIS-DAP Interface design that provides simple USB drag-n-drop programming and CMSIS-DAP debug interface for the target microcontroller.

NEXT: User Interfaces, Connectors, and Jumpers

PREVIOUS: Rebuild Firmware

5 User Interfaces, Connectors, and Jumpers

The following image shows where the connectors, headers, and jumpers are located on the Bambino 210.

Bambino 210 User Interfaces, Connectors, and Jumpers

5.1 Power Supply

The Bambino 210 can be powered from the USB0 device port on J4 or MBED HDK/USB1 device port on J6. The Bambino 210E can be powered from the USB0 device port, MBED HDK/USB1 device port, or the power jack (J1). A power mux automatically selects the power between USB0 and the MBED HDK/USB1. A FET is used to automatically select power from J1 should power be applied both the USB device ports and J1.

J1 comes standard with a 2.1 mm inner diameter and 5.5 mm outer diameter, positive center tapped female power supply jack. The minimum voltage that can be applied to J1 is 7 VDC and the maximum is 15 VDC. J1 can be changed to a 2 position screw terminal by desoldering the power jack and soldering in a screw terminal. A diode (D2) will protect the Bambino 210E should polarity of the power supply be reversed on the J1 connector. The protection diode is limited to a maximum of 1 amperes through it.

Figure x.x: Power supply connector configurations

5.2 Onboard Peripherals

5.2.1 USB0 Device

The Bambino 210 comes equipped with a USB Device Port. The Device port is compliant with the USB V2.0 high-speed device specification. It's connector is a micro USB Type AB.

USB0 Device Port	
Connector Pin#	**MCU Pin Name**
1	USB0_VBUS (+5.0V)
2	USB0_DM
3	USB0_DP
4	USB0_ID
5	Ground

5.2.2 MBED HDK/USB1 Device

The Bambino 210 comes equipped with the MBED HDK. It is powered by NXP's LPC11U35 microcontroller. The MBED HDK is compliant with the USB V2.0 full-speed device specification. It's connector is a micro USB Type AB and is designated J6. The LPC4330's USB1 is available on J6 when the MBED HDK is not populated through a custom configuration.

MBED HDK/USB1 Device Port		
Connector Pin#	**MBED HDK Pin (LPC11U35)**	**USB1 Pin (LPC4330)**
1	VUSB1	VUSB1
2	USB_DM	USB1_DM
3	USB_DP	USB1_DP
4	Not Connected	Not Connected
5	Ground	Ground

5.2.3 MBED HDK/CDC Device

The MBED HDK implements a virtual serial port via USB using I/O from UART2 in the LPC4330. This allows simple serial I/O from mbed applications to terminal emulators such as PuTTY.

HDK-CDC		
MCU Pin Name	**Peripheral**	**SCU Func**
P2_10	U2_TXD	2
P2_11	U2_RXD	2

5.2.4 Boot Jumper

The boot jumper is used to put the Bambino 210 into Device Firmware Upgrade (DFU) mode. This is accomplished by shorting the two pins before power is applied or by shorting the pins and pressing the reset button. For further information please see the Getting Started Section of this manual.

5.2.5 User Buttons and LEDs

The Bambino 210 comes standard with a user push button, a reset push button, and four user LEDs. The user push button is connected to GPIO0[7] with a 22k-ohm pull-up resistor connected to it. User LED1 (yellow) can be illuminated by clearing GPIO3[7] of the LPC4330. User LED2 (green) can be illuminated by clearing GPIO5[5]. User LED3 (blue) can be illuminated by clearing GPIO3[0]. User LED4 (red) can be illuminated by clearing GPIO3[1].

User Buttons and LEDs							
BAM210 Peripheral	**MCU Pin Name**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**
LED1	P6_11	GPIO3[7]	0	T2_MAT3	5		
LED2	P2_5	GPIO5[5]	4	T3_MAT2	6	USB0_IND0	7
LED3	P6_1	GPIO3[0]	0				
LED4	P6_2	GPIO3[1]	0				
BTN1	P2_7	GPIO0[7]	0				

5.2.6 Serial Flash Memory

The Bambino 210 uses serial flash for program and nonvolitile data storage. It uses the LPC43030's Quad SPI Flash Iinterface (SPIFI). The SPIFI interface has data rates up to 52 MB per second. The Bambino 210 comes standard with 4M of flash and the Bambino 210E comes standard with 8M of flash.

Serial Flash Memory		
MCU Pin Name	**Peripheral**	**SCU Func**
P3_3	SPIFI_SCK	3
P3_4	SPIFI_SIO3	3
P3_5	SPIFI_SIO2	3
P3_6	SPIFI_MISO	3
P3_7	SPIFI_MOSI	3
P3_8	SPIFI_CS	3

5.2.7 10/100 Ethernet (210E Only)

The Bambino 210E is equipped with a fully-integrated 10/100 Mbps Ethernet port. The Media Access Control (MAC) is implemented in the LPC4330 and the Physical (PHY) layer is implemented with Micrel?s KSZ8031. J3 is the RJ-45 connector and it has integrated magnetics and LEDs completes the Ethernet sub-system. Please see the KSZ8031 data sheet for further information on the PHY and the LPC4330 User's Manual for the MAC.

Ethernet		
MCU Pin Name	**Peripheral**	**SCU Func**
P0_0	ENET_RXD1	2
P0_1	ENET_TX_EN	6
P1_15	ENET_RXD0	2
P1_16	ENET_RX_DV	7
P1_17	ENET_MDIO	3
P1_18	ENET_TXD0	2

P1_19	ENET_REF_CLK	0
P1_20	ENET_TXD1	3
P2_0	ENET_MDC	7

5.2.8 MICRO SD (210E Only)

The microSD socket (J2) enables micro-secure-digital memory cards to be plugged into the Bambino 210E microcontroller board. The microSD card allows the user the ability of a standard removable media for transferring data to and from the Bambino 210E. The LPC4330 interfaces to the microSD card through the Secure Digital Input Output card interface.

Micro SD Card		
MCU Pin Name	**Peripheral**	**SCU Func**
CLK2	SD_CLK	4
P1_6	SD_CMD	7
P1_9	SD_DAT0	7
P1_10	SD_DAT1	7
P1_11	SD_DAT2	7
P1_12	SD_DAT3	7
P1_13	SD_CD	7

5.2.9 XBEE (210E Only)

The XBEE socket adds wireless support to the Bambino 210E. Digi International has several different versions of XBEE modules with different wireless protocols in the same physical footprint. Zigbee and WiFi are a couple of protocols supported by Digi International's XBEE modules. Please see Digi International's website for further details. The XBEE signals are shared with socket 5. Socket 5 should not be used if the XBEE module is being used.

XBEE		
MCU Pin Name	**Peripheral**	**SCU Func**
P5_6	U1_TXD	4
P1_14	U1_RXD	1
P5_2	U1_RTS	4
P5_4	U1_CTS	4
P5_1	GPIO2[10]	0

5.3 Expansion Headers

5.3.1 J7 Arduino Header

The Bambino 210's J7 female header is compatible with Arduino boards. The header uses 0.1 inch spacing between pins. The header contains power, ground, and the LPC4330's reset signal.

J7 Arduino Compatible Header	
PIN #	**Signal**

1	No Connection
2	GND
3	+3.3 VDC
4	nReset
5	+3.3 VDC
6	+5.0 VDC
7	GND
8	GND

5.3.2 J8 Arduino Header

The Bambino 210's J8 female header is compatible with Arduino boards. Some of the signals on J8 may be configured as analog inputs, an analog output, general purpose digital inputs or outputs (GPIO), or serial general purpose inputs and outputs (SGPIO).

J8 Arduino Compatible Header									
PIN #	**mbed**	**Arduino**	**MCU Pin Name**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**
1	p15	A0	P7_4	GPIO3[12]	0	ADC0_4			
2	p16	A1	P7_5	GPIO3[13]	0	CTOUT_12	1	ADC0_3	
3	p17	A2	P4_1	GPIO2[1]	0	CTOUT_1	1	ADC0_1	
4	p18	A3	P7_7	GPIO3[15]	0	CTOUT_8	1	ADC1_6	
5	p19	A4	P4_3	GPIO2[3]	0	SGPIO9	7	ADC0_0*	
6	p20	A5	P4_4	GPIO2[4]	0	SGPIO10	7	DAC*	
6**	p20	A5	ADC0_2						

(*) DAC and ADC0_0 use the same peripheral. If pin A5 is configured as a DAC, pin A4 should only be used for digital I/O.

(**) JP2 must be populated with a jumper on pins 1&2 and trace cut on bottom of board between pins 2&3

In order to maintain Arduino compatibility, pin 6 of J8 (A5) is hardwired to MCU pin P4_4 which can be a digital I/O or analog output. The following change can be performed for shields requiring an analog input on A5 please modify the board as follows:

- 1. Cut the trace between pins 2&3 on the bottom of the board.
- 2. Solder a 1x3 pin header into JP2.
- 3. Place a jumper on pins 1&2 of JP2.

Parts List for JP2				
Part Description	**Manufacturer**	**Part #**	**Digikey Part#**	**Mouser Part #**
1x3 pin header	Tyco	5-146280-3	5-146280-3	5-146280-3
shunt jumper	3M	969102-0000-DA	969102-0000-DA	969102-0000-DA

5.3.3 J9 Arduino Header

The Bambino 210's J9 female header is compatible with Arduino boards. Some of the signals on J9 may be configured as a UART, PWM, or general purpose digital inputs or outputs (GPIO).

J9 Arduino Compatible Header

PIN #	mbed	Arduino	MCU Pin Name	Peripheral	SCU Func	Peripheral	SCU Func
1	p21	D0	P6_5	GPIO3[4]	0	U0_RXD	2
2	p22	D1	P6_4	GPIO3[3]	0	U0_TXD	2
3	p23	D2	P1_7	GPIO1[0]	0	CTOUT_13	2
4	p24	D3	P4_0	GPIO2[0]	0		
5	p25	D4	P6_9	GPIO3[5]	0		
6	p26	D5	P5_5	GPIO2[14]	0		
7	p27	D6	P5_7	GPIO2[7]	0		
8	p28	D7	P7_6	GPIO3[14]	0	CTOUT_11	1

5.3.4 J10 Arduino Header

The Bambino 210's J10 female header is compatible with Arduino boards. Some of the signals on J10 may be configured as a UART, PWM, I2C, SGPIO, or GPIO.

J10 Arduino Compatible Header									
PIN #	**mbed**	**Arduino**	**MCU Pin Name**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**
1	p29	D8	P6_12	GPIO2[8]	0	CTOUT_7	1		
2	p30	D9	P5_0	GPIO2[9]	0	MCOB2	1		
3	p31	D10	P4_6	GPIO2[6]	0	CTOUT_4	1	SGPIO12	7
4	p32	D11	P4_8	GPIO5[12]	4	SGPIO13	7		
5	p33	D12	P4_9	GPIO5[13]	4	SGPIO14	7		
6	p34	D13	P4_10	GPIO5[14]	4	SGPIO15	7		
7	GND								
8	AREF								
9	p37	I2C1_SDA	P2_3	GPIO5[3]	4	I2C1_SDA	1	U3_TXD	2
10	p38	I2C1_SCL	P2_4	GPIO5[4]	4	I2C1_SCL	1	U3_RXD	2

5.3.5 J11 Extended I/O Header (BAM210E)

The Bambino 210E's J11 extended I/O header's signals may be configured as SGPIO, or GPIO.

J11 Extended I/O Header						
PIN #	**mbed**	**MCU Pin Name**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**
1	p47	P6_3	GPIO3[2]	0	SGPIO4	2
2	p48	P6_6	GPIO0[5]	0	SGPIO5	2
3	p49	P6_7	GPIO5[15]	4	SGPIO6	2
4	p50	P6_8	GPIO5[16]	4	SGPIO7	2
5	GND					

6	3.3 VDC			
7	p53	P2_2	GPIO5[2]	4
8	p54	P2_1	GPIO5[1]	4

5.3.6 J12 Extended I/O Header (BAM210E)

The Bambino 210E's J12 extended I/O header's pin 1 and 2 are only Analog inputs. Pins 3 and 4 may be configured as GPIO or SGPIO. Pins 5 and 6 are strictly GPIO.

J12 Extended I/O Header								
PIN #	**mbed**	**MCU Pin Name**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**
1	p55	ADC0_5	ADC1_5					
2	p56	ADC0_7	ADC1_7					
3	p57	P2_6	GPIO5[6]	4	SGPIO7	0		
4	p58	P2_8	GPIO5[7]	4	SGPIO15	0	CTOUT_0	1
5	p59	P6_10	GPIO3[6]	0				
6	p60	P2_9	GPIO1[10]	0	CTOUT_3	1		

5.3.7 J13 Extended I/O Header (BAM210E)

Some of the Bambino 210E's J13 extended I/O header's signals may be configured as GPIO or SGPIO. .

J13 Extended I/O Header								
PIN #	**mbed**	**MCU Pin Name**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**
1	p61	P7_3	GPIO3[11]	0				
2	p62	P3_2	GPIO5[9]	0				
3	p63	P7_2	GPIO3[10]	0	SGPIO6	7		
4	p64	P3_1	GPIO5[8]	4				
5	p65	P7_1	GPIO3[9]	0	CTOUT_15	1	SGPIO5	7
6	p66	P7_0	GPIO3[8]	0	CTOUT_14	1	SGPIO4	7
7	p67	P4_2	GPIO2[2]	0	SGPIO8	7		
8	p68	P4_5	GPIO2[5]	0	SGPIO11	7		

5.3.8 J14 Extended I/O Header (BAM210E)

Some of the Bambino 210E's J14 extended I/O header's signals may be configured as PWM, SGPIO, GPIO, or SPI. Pin 10 can only be configured as the SPI clock.

J14 Extended I/O Header								
PIN #	**mbed**	**MCU Pin Name**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**
1	p69	P2_13	GPIO1[13]	0				
2	p70	P2_12	GPIO1[12]	0				

3	p71	P9_6	GPIO4[11]	0	MCOB1	1	SGPIO8	6
4	p72	P9_5	GPIO5[18]	4			SGPIO3	6
5	p73	P5_3	GPIO2[12]	0				
6	p74	P1_8	GPIO1[1]	0				
7	p75	P1_5	GPIO1[8]	0	CTOUT_10	1	SSP1_SSEL	5
8	p76	P1_4	GPIO0[11]	0	SSP1_MOSI	5		
9	p77	P1_3	GPIO0[10]	0	SSP1_MISO	5		
10	p78	PF_4			SSP1_SCK	0		

5.4 Field Installable Options

The coin battery holder on the bottom of the Bambino 210 is not populated at production time. The Cortex M JTAG is also not populated when the board is built. Both parts may be purchased from DigiKey and Mouser.

Field Installable Options Parts List				
Option	**Manufacturer**	**Part #**	**Digikey Part#**	**Mouser Part #**
Coin Battery Holder	Keystone	3002	3002K-ND	534-3002
Coin Battery	Panasonic	CR2032	P189-ND	658-CR2032
Cortex M JTAG	FCI	20021521-00010T1LF	609-4054-ND	649-200215210010T1LF
J15 (2x5 pin header)	Tyco	5-146256-5	5-146256-5	5-146256-5
J16 (1x6 pin header)	Harwin	M20-9990646	M20-9990646	M20-9990646

5.4.1 Coin Battery

The Bambino 210's microcontroller has a built in real-time clock calendar that can be battery backed by supplying 2.2 VDC to 3.6 VDC to the VBAT pin on the LPC4330. A battery holder can be added to the bottom of the board to power the VBAT pin with a coin cell battery. The battery holder is manufactured by Keystone and it's part number is 3002. The battery holder accepts CR2032 series coin cells. Power is only drawn from the battery when the power is off to the Bambino 210.

5.4.2 Cortex M JTAG

A JTAG port (J5) can be added for software download and debugging. The JTAG port allows users to set break points and to single step through their program. For detailed information on the operation of the JTAG port and TAP controller, please refer to IEEE Standard 1149.1-Test Access Port and Boundary-Scan Architecture.

Cortex M JTAG	
Connector Pin#	**Pin Name**
1	VCC (+3.3V)
2	TMS/SWDIO
3	Ground
4	TCK/SWDCLK
5	Ground
6	TDO/SWO

7	No Connect
8	TDI
9	Ground
10	RESET

5.4.3 J15 PMOD-I2C Header (Unpopulated)

Pmods? are small I/O interface boards that offer an ideal way to extend the capabilities of the Bambino 210. Pmod is trade marked by Digilent Inc. J15 is set-up as an I2C Pmod header. The Bambino 210 comes with the connector unpopulated.

J15 PMOD-I2C Header	
PIN #	**MCU Pin Name**
1	I2C0_SCL
2	I2C0_SCL
3	I2C0_SDA
4	I2C0_SDA
5	GND
6	GND
7	3.3 VDC
8	3.3 VDC

5.4.4 J16 PMOD-SSP Header (Unpopulated)

J16 is set-up as an SPI/UART Pmod header. The Bambino 210 comes with the connector unpopulated.

J16 PMOD-SSP Header								
PIN #	**mbed**	**MCU Pin Name**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**	**Peripheral**	**SCU Func**
1	p80	P1_0	GPIO0[4]	0	SSP0_SSEL	5		
2	p81	P1_2	GPIO0[9]	0	SSP0_MOSI	5	SGPIO9	3
3	p82	P1_1	GPIO0[8]	0	SSP0_MISO	5	SGPIO8	3
4	p83	P3_0			SSP0_SCK	4		
5	GND							
6	3.3 VDC							

NEXT: Mechanical and Electrical Characteristics

PREVIOUS: Hardware

6 Mechanical and Electrical Characteristics

6.1 Absolute Minimum and Maximum Ratings

Characteristic	Minimum	Maximum	Unit
Voltage on J1	7.0	15.0	VDC
Voltage on VBAT (Coin Cell Battery Holder)	0.0	3.3	VDC
Voltage on ADC	0.0	3.3	VDC
Voltage on Digital Input	0.0	5.0	VDC
Operating Temperature	0	70	ºC
Storage Temperature	-50	125	ºC

The Bambino 210 is currently available for commercial temperature ranges. Contact the Micromint sales department if you require support for industrial temperature ranges.

6.2 Mechanical Dimensions

Below is the physical dimensions for the Bambino 210. The mounting holes will accept a #4 size screw.

DIM	Inches	Millimeters	DIM	Inches	Millimeters	DIM	Inches	Millimeters	DIM	Inches	Millimeters
A	4.0	101.6	**C**	0.1	2.54	**E**	1.11	28.194	**G**	0.2	5.08
B	1.83	46.482	**D**	0.61	15.494	**F**	2.3	58.42	**H**	0.91	23.114

Lincoln 60 Mechanical Dimensions

DIM	Inches	Millimeters	DIM	Inches	Millimeters	DIM	Inches	Millimeters	DIM	Inches	Millimeters
A	0.33	8.4	**F**	0.63	16.0	**K**	0.136	3.45	**P**	0.3	7.62
B	0.3543	9.0	**G**	0.47	11.94	**L**	0.545	13.843	**Q**	0.335	8.5
C	0.105	2.67	**H**	0.269	6.83	**M**	0.86	21.84	**R**	0.173	4.4
D	0.345	8.763	**I**	0.177	4.5	**N**	0.56693	14.40	**S**	0.1	2.54
E	0.065	1.65	**J**	0.256	6.5	**O**	0.225	5.715	**T**	0.06	1.524

Lincoln 60 Suggested Openings

NEXT: References

PREVIOUS: User Interfaces, Connectors, and Jumpers

7 References

This section outlines material that may be useful for further reading.

7.1 Documents

mbed Texbook Overview
http://mbed.org/cookbook/Textbook
Overview of the textbook "Fast and Effective Embedded Systems Design: Applying the ARM mbed" by Rob Toulson and Tim Wilmshurst.

mbed Texbook Course Notes
http://mbed.org/cookbook/Course-Notes
Course notes for textbook referenced above.
LPC43XX User Manual
http://www.nxp.com/documents/user_manual/UM10503.pdf
This user manual provides reference information for the NXP LPC43XX microcontrollers. All MCU registers are documented in the user manual.

7.2 Books

Fast and Effective Embedded Systems Design: Applying the ARM mbed

ISBN: 0080977685 Publisher: Newnes; (August, 2012)

Introduction to embedded systems design, using the ARM mbed and C programming language as development tools.

The Definitive Guide to ARM® Cortex-M3 and Cortex-M4 Processors, Third Edition

by Joseph Yiu

ISBN: 0124080820 Publisher: Newnes (November, 2013)

Overview of the processor and instruction set architecture of the ARM® Cortex®-M3 and Cortex®-M4 processors. Several code examples using IAR, Keil, gcc and CooCox CoIDE.

The Designer's Guide to the Cortex-M Processor Family: A Tutorial Approach

by Trevor Martin

ISBN: 0080982964 Publisher: Newnes (May, 2009)

Tutorial-based book giving the key concepts required to develop programs in C with a Cortex M- based processor.

ARM System Developer's Guide: Designing and Optimizing System Software

ISBN: 1558608745 Publisher: Morgan Kaufmann; (March, 2004)

In-depth overview of the ARM architecture with examples that outline impact of programming practices on performance, power and cost.

7.3 Useful Web Links

Micromint Web Site
http://www.micromint.com/
Product information and software updates for the Electrum SBCs.

NXP?s Web Site
http://www.nxp.com/
Manuals, Erratas, and application notes for NXP MCUs
mbed Community
http://mbed.org/
mbed community site with documentation, code examples and forums

---

NEXT: Appendix A - Updating CMSIS-DAP Firmware

PREVIOUS: Mechanical and Electrical Characteristics

8 Appendix A

8.1 Appendix A - Updating CMSIS-DAP Firmware

- 1. Locate the updated CMSIS-DAP binary file
- 2. Boot the MBED HDK into CRP DISABLED mode by shorting JP3 while connecting the USB1 Cable. A

drive should appear.

- 3. Program the Interface firmware by deleting the firmware.bin file and copying the new binary to the CRP DISABLED disk.
- 4. Reboot the Interface by disconnecting and reconnecting the USB1 cable without JP3 shorted
- 5. Verify that the MBED disk comes up
- 6. Finished!

NEXT: Appendix B - Using an External JTAG with the BAM210

PREVIOUS: References

9 Appendix B

9.1 Appendix B - Using an External JTAG with the BAM210

9.1.1 Configuring the Bambino 210 to use an External JTAG

The JTAG connector is optional for the BAM210 and must be populated in order to use an external JTAG.

Follow these instructions to configure the Bambino 210 to use an external JTAG.

- 1. Locate the LPC11U35 Virtual COM Port Binary
- 2. Boot the MBED HDK into CRP DISABLED mode by shorting JP3 while connecting the USB Cable to USB1. A

drive should appear.

- 3. Program the Interface firmware by deleting the firmware.bin file and copy lpc11u35_vcom.bin to the CRP DISABLED disk.
- 4. Reboot the Interface by disconnecting and reconnecting the USB1 cable without JP3 shorted.
- 5. Finished!

The Bambino 210 is now ready to use with an external JTAG. A virtual COM port is now available to use on USB1.

The LPC4330 signals used for the virtual COM port are listed below:

LPC4330 PIN NAME	MBED ALIAS
P2_10	USBTX
P2_11	USBRX

Signals Used for Virtual COM Port

9.1.2 Configuring Keil µVision Compiler

Please follow these steps for setting up Keil µVision compiler for debugging the Bambino 210's LPC4330 microcontroller.

- 1. Press ALT and F7 on the keyboard to bring up the Options for Target window.
- 2. Click on the "Debug" tab.
- 3. Choose the debugger being used from the drop down in the upper right hand side of the window.
- 4. Click on "Settings" next to where the debugger was selected to bring up the Target Driver Setup window.
- 5. Click on the "Flash Download" tab.

- 6. Click the "Add" button.
- 7. Scroll and select "LPC18xx/43xx S25FL032 SPIFI".
- 8. Change the "Size" to 0x4000 and the Target Driver Setup window should look like the image below.

NEXT: Appendix C - Configuring Keil MDK for CMSIS-DAP

PREVIOUS: Appendix A - Updating CMSIS-DAP Firmware

10 Appendix C

10.1 Appendix C - Configuring Keil MDK for CMSIS-DAP

Please follow these steps for setting up Keil µVision compiler for debugging the Micromint Bambino using the mbed HDK with CMSIS-DAP firmware.

- 1. Press ALT and F7 on the keyboard to bring up the Options for Target window.
- 2. Click on the "Debug" tab.
- 3. Choose the "CMSIS_DAP Debugger" from the drop down in the upper right hand side of the window. Specify the location of the LPC43xx_spifi.ini initialization file.

- 4. Click on "Settings" next to where the debugger was selected to bring up the Target Driver Setup window. Select "JTAG" for multi-core debugging.

- 5. Click on the "Flash Download" tab.
- 6. Click the "Add" button.
- 7. Scroll and select "LPC18xx/43xx S25FL032 SPIFI".
- 8. Change the "Size" to 0x4000 and the Target Driver Setup window should look like the image below.

11 Appendix D

11.1 DFU Firmware Updates

The Bambino 210 firmware can be updated via port USB0 using the standalone NXP DFU Flash Programmer. This tool is an alternative to update firmware when your LPC4330 is in an unknown state or generates an ARM hard fault. For most other cases, using the mbed virtual flash drive should be simpler. To update firmware via DFU, please follow these steps:

1. Install the DFU Flash Programmer to a directory in your hard disk. Currently the NXP DFU programmer is only available for Windows.

2. Connect a cable to port USB0 and place the board in USB boot mode by shorting the two contacts labelled "Boot JP1" as shown below while pressing and releasing the RESET button. Metal tweezers work great for shorting the two contacts. If you will be doing frequent DFU updates, you may consider soldering a 2-pin header in and use a jumper to enter the USB boot mode.

Boot Jumper

3. An entry "LPC USB" should appear in the Windows Device Manager. If your device is not recognized, please check that you have the USB Drivers installed.

LPC USB device

4. Run lpc_dfutil.exe in the dfusec folder. You should see "HIGH SPEED USB" in the status bar indicating it was able to connect to the board.

LPC DFU Flash Programmer

5. Use the following parameters and press START. You may need to use full paths for the algorithm (*.hdr) and firmware (*.bin) files.

```
Algorithm: .\Programming_algorithms\iram_dfu_util_spiflash.bin.hdr
File: <Path and name of firmware to be copied to flash>
Address: 0x14000000
Size:    0x00400000
Param:   0x00000000
Device erase: Region
Operation after: Reset
No checkboxes neede
```

You can test the procedure using the simple blinky binary listed below.

.\Prebuilt_examples\bambino210\BAM210 Blinky.bin
blinky.bin

6. After the flash is complete, exit the utility, remove the boot jumper, and reset your board.

PREVIOUS: Appendix C - Configuring Keil MDK for CMSIS-DAP