Intro to Cortex M0 and LPCxpresso 1114

Minute Quiz

Minute Quiz

- Just kidding, but...
- What are the steps to go from source code to an executable?
  - Compile
  - Link

Compile

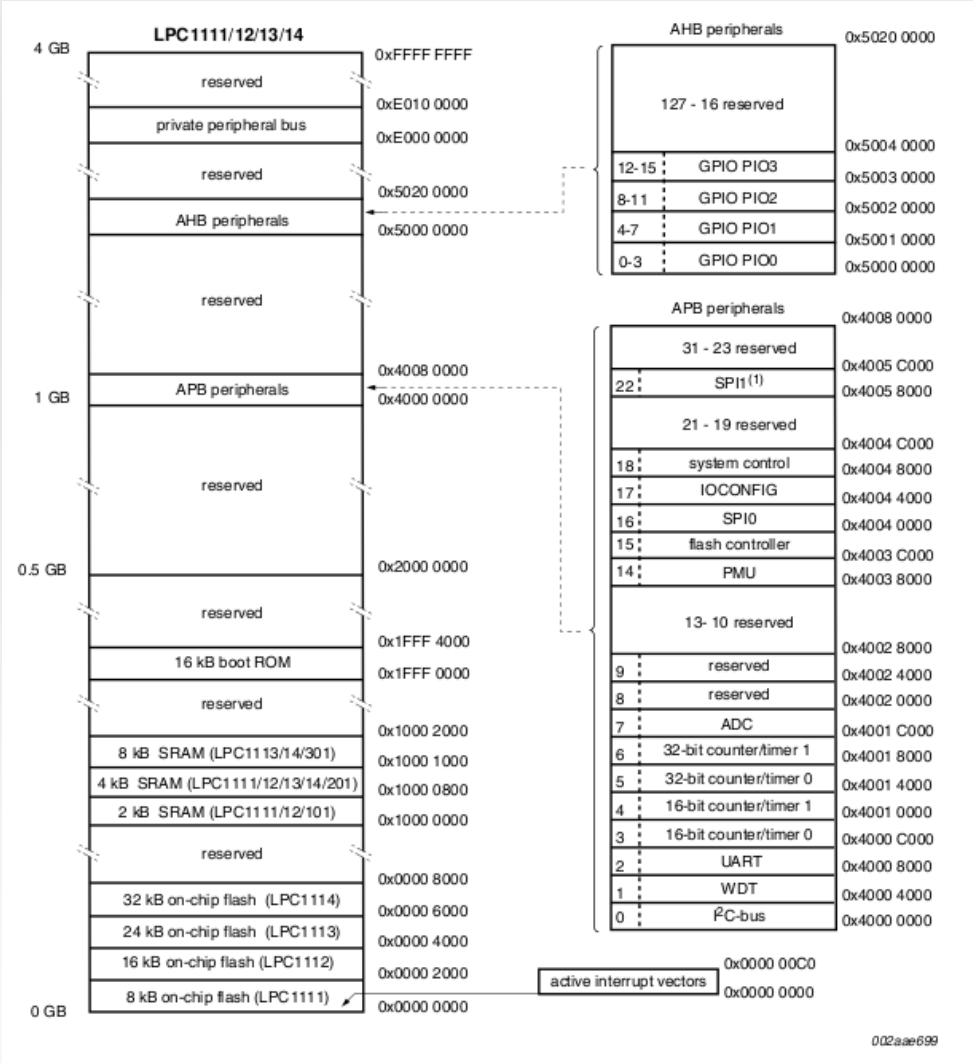
- What makes ”gcc” different from ”arm-none-eabi-gcc”?
- What does the ”-mach” flag do to gcc?
- All ARM assembly is the same, right?
  - So why can't I run code for the SmartFusion on my ARM-based Android phone?

Linking

- What is linking?
- We added this to every SoftConsole, what is it?
  - -T ../CMSIS/startup_gcc/debug-in-actel-smartfusion-esram.ld

Linking

- What is linking?
- We added this to every SoftConsole, what is it?
  - -T ../CMSIS/startup_gcc/debug-in-actel-smartfusion-esram.ld



Static vs Dynamic Linking

- What does ”#include <stdlib.h>” do?
  - Compiler?
  - Linker?
- Where is stdlib.o?
  - On your laptop
    - /usr/lib/libc.so
  - On your SmartFusion?

Vocabulary

- Cross-compiler
  - arm-none-eabi-gcc
  - arm-none-eabi-clang
- Toolchain
  - compiler + linker + supporting scripts / configuration

Background

- ARM
  - Cortex M0
    - LPCxpresso 114
      - Peripherals
      - LPCxpresso (development suite)
      - CooCox (OS)
      - TinyOS

ARM

- 32-bit RISC ISA
  - With 16-bit subset: Thumb
- Simple, efficient cores
- Dominant in mobile / embedded space
- Trivia (ARM vs. ARM Holdings)
  - The acronym ARM originally stood for Acorn RISC Machine. The company name ARM stands for Advanced RISC Machines. This name was changed, around the time of the IPO, to "ARM Holdings", since it was felt the term RISC, which indicates a type of CPU design, being phonetically identical to "risk," would deter people unfamiliar with computers. [wikipedia]

ARM: Cortex Family

- Cortex == ARMv7
- 3 ”Families”
  - Cortex-**A**: Applications
    - Smartphones, etc
  - Cortex-**R**: Real-Time
  - Cortex-**M**: Microcontrollers

ARM: Cortex M Family

- Cortex-M4
  - ARMv7-ME
    - Thumb, Thumb2, FPU. Hardware MAC, SIMD, and divide
- Cortex-M3
  - ARMv7-M
    - Thumb2, hardware divide
- Cortex-M0
  - ARMv6-M
    - Thumb2 subset (16-bit Thumb instructions & BL, MRS, MSR, ISB, DSB, and DMB(16-bit Thumb instructions & BL, MRS, MSR, ISB, DSB, and DMB)

Cortex M0

- Simplest, smallest ”current generation” ARM
- 85 µW/MHz (0.085 milliWatt)
- 12K gates
- Only 56 instructions
  - Subset of M3/M4; Thumb and some Thumb2
- 3-stage pipeline
- Interrupts: NMI + 1-32 physical interrupts
  - 16 cycle latency
- Complex Hardware Ops
  - Single-Cycle 32x32 multiply

LPCxpresso 1114 Specs

- Cortex-M0 @ 50 Mhz (max)
- 32kB Flash
- 8kB RAM
- 12.000 MHz clock crystal
- Timers:
  - 4 capture inputs, 13 match outputs
  - Two 32-bit counter/timers
  - Two 16-bit counter/timers
  - One Programmable Watchdog timer

LPCxpresso 1114 Specs

- Clocks:
  - 12 MHz RC oscillator, 1% accuracy
  - Crystal operator, ranged 1-25MHz
    - PLL allows CPU frequency up to 50MHz
  - Watchdog oscillator, ranged 7.8KHz-1.8MHz
  - Clock output with divider can source any clock

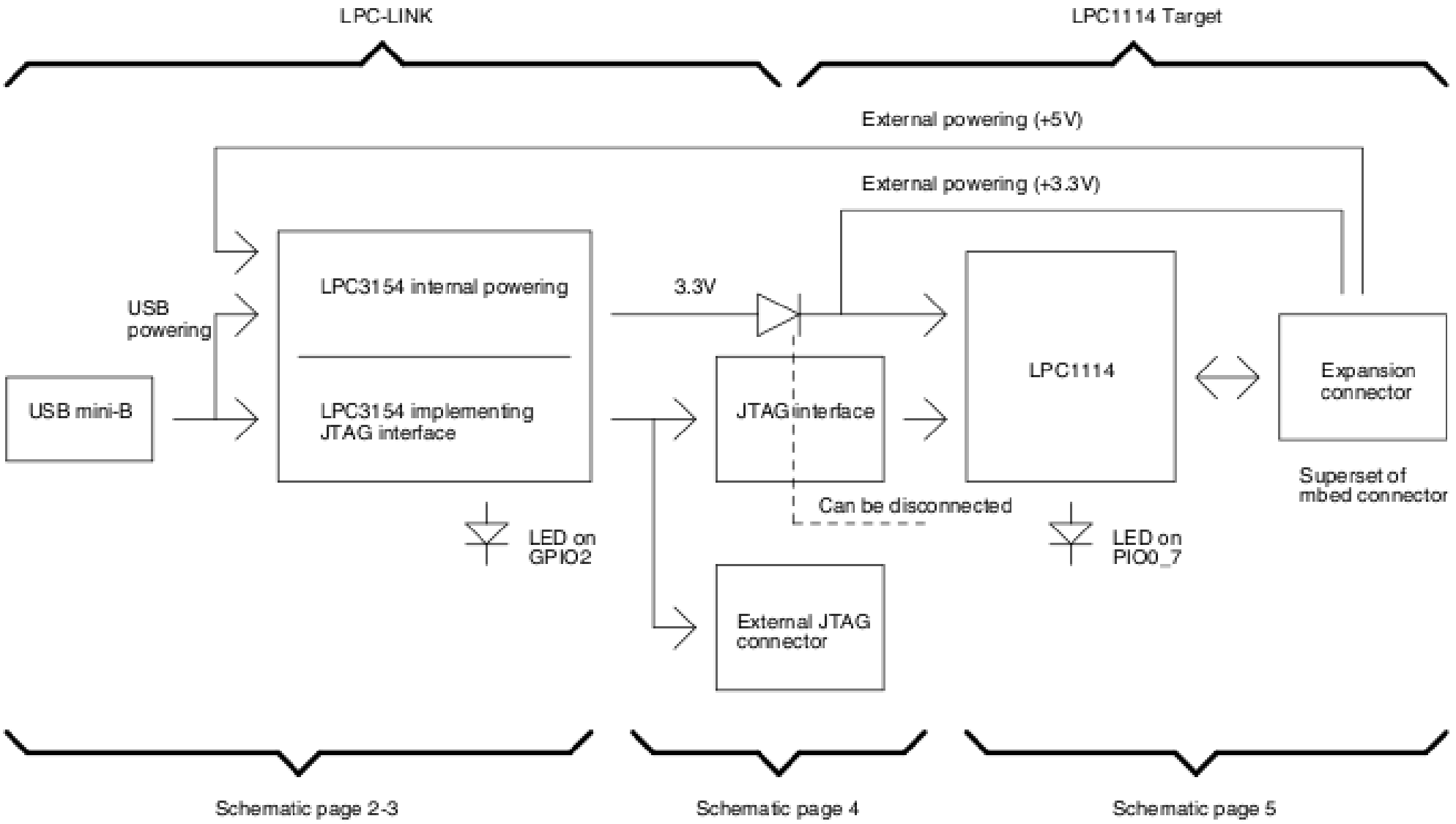
LPCxpresso 1114 Specs

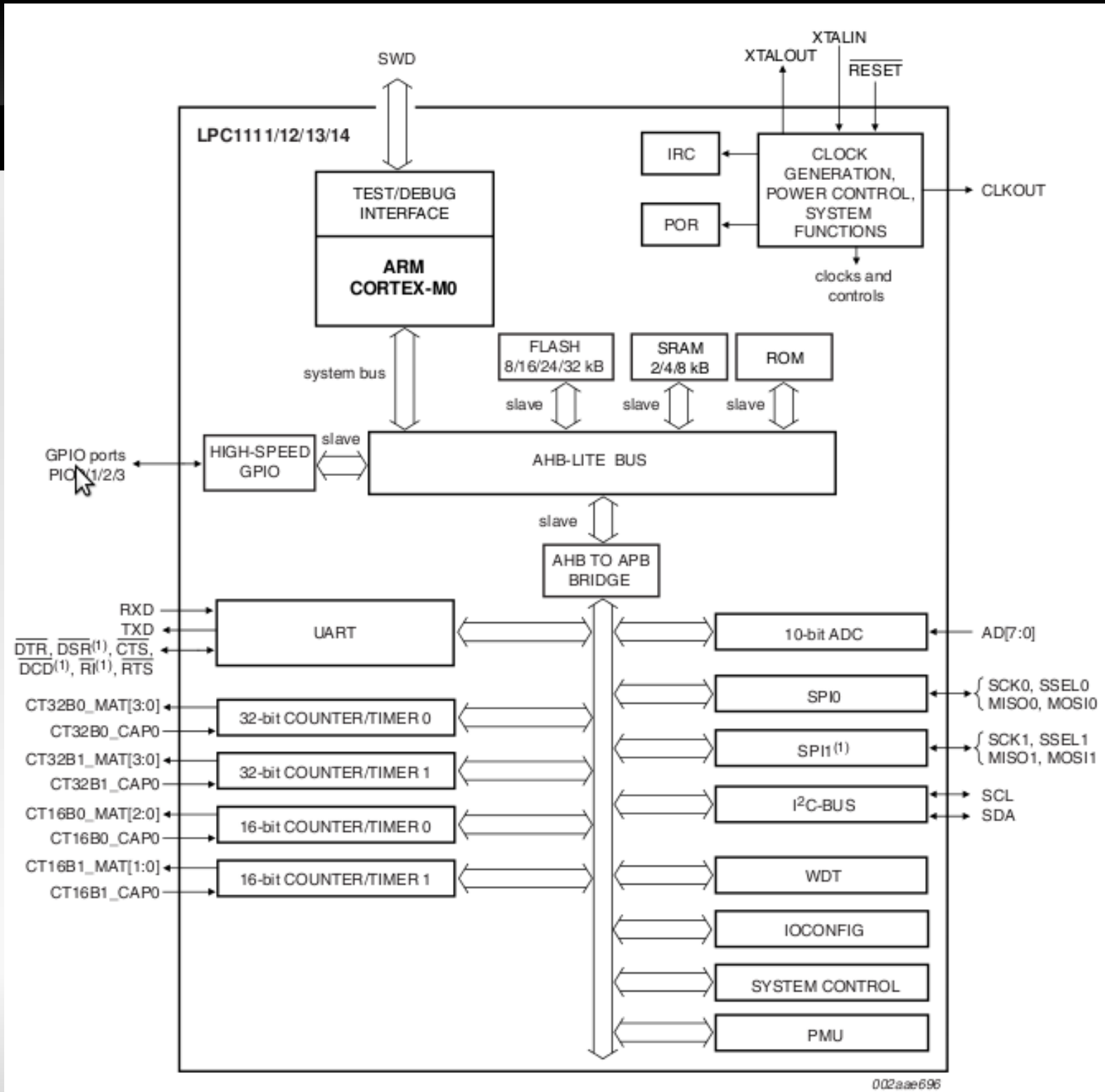
- 42 GPIOs with configurable pull-up/down resistors
- Any GPIO usable as edge/level triggered interrupt
- High-current output driver (20 mA) on one pin.
- High-current sink drivers (20 mA) on two I2C-bus pins in Fast-mode Plus.
- Four general purpose counter/timers with a total of four capture inputs and 13 match outputs.

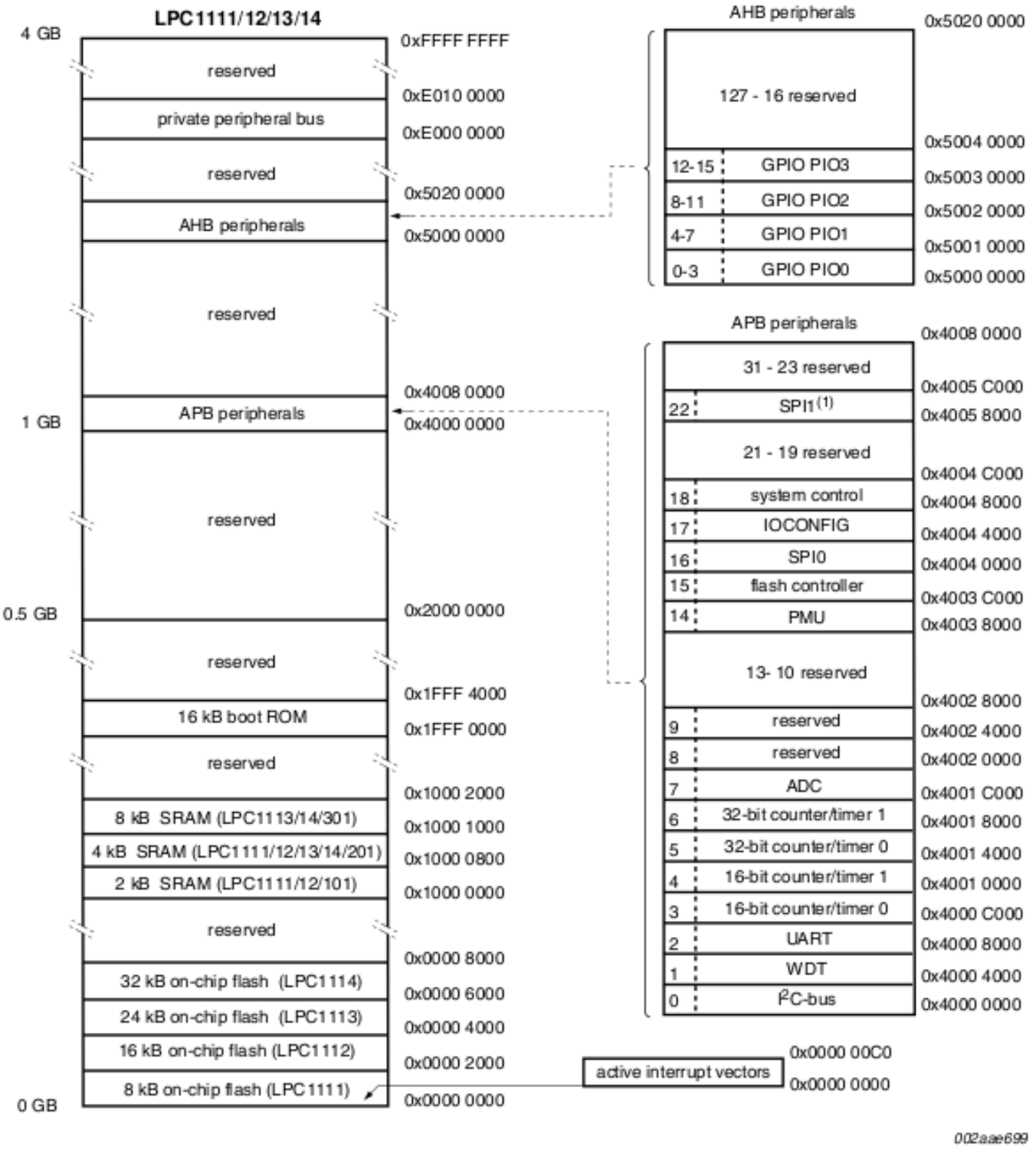
LPCxpresso 1114 Specs

- Analog:
  - 10-bit ADC
    - single 10-bit successive approximation ADC with eight channels
    - Measurement range 0 V to VDD.
    - 10-bit conversion time ≥ 2.44 µs.
- Serial:
  - UART
  - 1 SPI with SSP support
  - I2C with FastMode (up to 1 Mbit/s)

LPCxpresso 1114




SWD
XTALIN
XTALOUT
RESET
LPC1111/12/13/14
IRC
CLOCK GENERATION, POWER CONTROL, SYSTEM FUNCTIONS
CLKOUT
POR
clocks and controls
TEST/DEBUG INTERFACE
ARM CORTEX-M0
system bus
FLASH 8/16/24/32 kB
SRAM 2/4/8 kB
ROM
slave
slave
slave
GPIO ports PIO0/1/2/3
HIGH-SPEED GPIO
slave
AHB-LITE BUS
slave
AHB TO APB BRIDGE
RXD
TXD
DTR, DSR(1), CTS, DCD(1), RI(1), RTS
UART
10-bit ADC
AD[7:0]
SPI0
SCK0, SSEL0 MISO0, MOSI0
SPI1(1)
SCK1, SSEL1 MISO1, MOSI1
I2C-BUS
SCL
SDA
CT32B0_MAT[3:0]
CT32B0_CAP0
32-bit COUNTER/TIMER 0
CT32B1_MAT[3:0]
CT32B1_CAP0
32-bit COUNTER/TIMER 1
CT16B0_MAT[2:0]
CT16B0_CAP0
16-bit COUNTER/TIMER 0
CT16B1_MAT[1:0]
CT16B1_CAP0
16-bit COUNTER/TIMER 1
WDT
IOCONFIG
SYSTEM CONTROL
PMU
002aae696


LPC1111/12/13/14
4 GB
1 GB
0.5 GB
0 GB
0xFFFF FFFF
reserved
0xE010 0000
private peripheral bus
0xE000 0000
reserved
0x5020 0000
AHB peripherals
0x5000 0000
reserved
0x4008 0000
APB peripherals
0x4000 0000
reserved
0x2000 0000
reserved
0x1FFF 4000
16 kB boot ROM
0x1FFF 0000
reserved
0x1000 2000
8 kB SRAM (LPC1113/14/301)
0x1000 1000
4 kB SRAM (LPC1111/12/13/14/201)
0x1000 0800
2 kB SRAM (LPC1111/12/101)
0x1000 0000
reserved
0x0000 8000
32 kB on-chip flash (LPC1114)
0x0000 6000
24 kB on-chip flash (LPC1113)
0x0000 4000
16 kB on-chip flash (LPC1112)
0x0000 2000
8 kB on-chip flash (LPC1111)
0x0000 0000
AHB peripherals
0x5020 0000
127 - 16 reserved
0x5004 0000
12-15 GPIO PIO3
0x5003 0000
8-11 GPIO PIO2
0x5002 0000
4-7 GPIO PIO1
0x5001 0000
0-3 GPIO PIO0
0x5000 0000
APB peripherals
0x4008 0000
31 - 23 reserved
0x4005 C000
22 SPI1 (1)
0x4005 8000
21 - 19 reserved
0x4004 C000
18 system control
0x4004 8000
17 IOCONFIG
0x4004 4000
16 SPI0
0x4004 0000
15 flash controller
0x4003 C000
14 PMU
0x4003 8000
13- 10 reserved
0x4002 8000
9 reserved
0x4002 4000
8 reserved
0x4002 0000
7 ADC
0x4001 C000
6 32-bit counter/timer 1
0x4001 8000
5 32-bit counter/timer 0
0x4001 4000
4 16-bit counter/timer 1
0x4001 0000
3 16-bit counter/timer 0
0x4000 C000
2 UART
0x4000 8000
1 WDT
0x4000 4000
0 I2C-bus
0x4000 0000
active interrupt vectors
0x0000 00C0
0x0000 0000
002aae699

Let there be Light!

- (emitted in a blinking pattern from diodes)
- Blinky
  - Set a 10ms timer

```
void TIMER32_0_IRQHandler(void)

{

LPC_TMR32B0->IR = 1;/* clear
interrupt flag */

timer32_0_counter++;

return;

}
```

And loop forever:

```
while (1) {

   /* Each time we wake up... */

   /* Check TimeTick to see whether to set or
      clear the LED I/O pin */

   if ( (timer32_0_counter%LED_TOGGLE_TICKS) <
        (LED_TOGGLE_TICKS/2) )

   {

      GPIOSetValue( LED_PORT, LED_BIT, LED_OFF );

   } else {

      GPIOSetValue( LED_PORT, LED_BIT, LED_ON );

   }

   /* Go to sleep to save power between timer
   interrupts */

   __WFI();

}
```

OS'es for Cortex M0

- Linux Kernel?
- make allnoconfig
  - With some editing to target the closest NXP board
  - And LZMA (best, slowest) compression...
- ls -lh
  - 943K Image
  - 343K zImage
- 32k of flash...

TinyOS

- Event-driven, non-preemptable
  - Except for the thread library
- More extenisive networking stack
- Doesn't support M0 'out of the box'
  - *”There is work underway to support the Cortex M3”*

CooCox CoOS

- ”CooCox CoOS is an embedded real-time multi-task OS specially for ARM Cortex M series.”
  - Scalable, minimum system kernel is only 974Bytes
  - Supports preemptive priority and round-robin
  - Interrupt latency is 0
  - Stack overflow detection option
  - Semaphore, Mutex, Flag, Mailbox and Queue for communication & synchronisation
- *http://www.coocox.org/CoOS.htm*

Programming the 1114 (on Linux)

- LPCxpresso tools installed and working
  - Build, run, debug, etc
- CoIDE and tools installed and mostly working
  - Everything except CoFlash works
  - But can flash from command line with LPC tools...
    - crt_emu_lpc11_13_nxp -g -mi -2 -pLPC1114/301 -wire=winUSB -flash-load-exec=blinky.axf
- Demo IDEs

Questions?