Network Embedded Systems
Sensor Networks – Fall 2013

Hardware

Marcus Chang, mchang@cs.jhu.edu
Embedded Systems

- Designed to do one or a few dedicated and/or specific functions

- *Embedded* as part of a complete device often including hardware and mechanical parts

- By contrast, a general-purpose computer, is designed to be flexible and to meet a wide range of end-user needs

- Wikipedia
Personal Computers

- Individual Components
  - CPU
  - Northbridge
    - RAM
    - GPU
  - Southbridge
    - Hard drive
    - WIFI
    - USB

Microcontroller Unit (MCU)

- Multiple components in same chip
  - CPU
  - Volatile memory (e.g. RAM)
  - Persistent memory (e.g. Flash)
  - Analog-to-Digital Converters
  - Timers (alarms, stop-watches)
  - Peripheral interfaces
  - Radio
  - ...

- Think of multiple cores, some with only one purpose but very high efficiency/low power consumption

Image: Texas Instruments
System on a Chip (SoC)

- Microcontrollers are systems on a single chip
  - However, often people think of SoC as more powerful than MCU

- Qualcomm Snapdragon

![Qualcomm Snapdragon Diagram](Image: Qualcomm)
Famous Microcontrollers / System on a Chip

- Home appliances, Auto industry, Industrial systems
  - Intel 8051 – billions units shipped per year (2008)
  - Microchip PIC – 6 billion in total (2008)
  - Atmel AVR – (Arduino)
  - Texas Instruments MSP430

- Cell phones, Tablets
  - ARM Cortex
    - Apple A6
    - Qualcomm Snapdragon
    - Samsung Exynos
Telos rev. B, Telosb, Tmote Sky
Telos rev. B, Telosb, Tmote Sky

- 2-pin SVS connector
- Texas Instruments MSP430 F1611 microcontroller
- 48-bit silicon serial ID
- USB Flash (2kB)
- 32kHz oscillator
- ST Code Flash (1MB)
Telosb Block Diagram

- PCB Antenna
- SMA Coax
- CC2420 Radio 2.4 GHz IEEE 802.15.4 compliant
- SPI I/O
- Silicon Serial ID 1-wire
- TI MSP430 Microcontroller
  - SPI[0]
  - UART[1] P1[0,3,4] P4[1,5,6]
  - UART[0]
  - I2C[0]
  - ADC[0-3,6-7]
  - GPIO
  - Reset
  - User
  - SVSin
  - SVSout
- 10-pin + 6-pin IDC header
- ST Flash 1024k (2.7V)
- Write Protection
- RX/TX RTS/DTR
- USB 2.0 UART/RS232 Functionality
- JTAG 8-pin 2mm IDC header
- Power
- I/O
- ADC[4]
- ADC[5]
- Humidity Temperature Sensor
- PAR Sensor
- TSR Sensor
Texas Instruments MSP430F1611 MCU
MSP430 System Clock

- Drives everything

- Inputs:
  - Low-frequency crystal (kHz range)
  - High-frequency crystal (MHz range)
  - Digitally-Controlled Oscillator (DCO)

- Outputs:
  - Master Clock
  - Sub-Main Clock
  - Auxiliary Clock

Image: Hubert Hagadorn
Telosb Clock Sources

- **External Sources**
  - Crystal (32 kHz)

- **Precision: Crystal > DCO**
  - Calibrate DCO with crystal

- **Temperature:**
  - Crystal, DCO change frequency
  - Store temperature specific calibration parameters

Image: Dallas Semiconductors
System Clock Configuration

- **Input-to-Output**
  - User configurable with constraints

- **CPU**
  - Master System Clock

- **Peripherals**
  - Master System
  - Sub-main Clock
  - Auxiliary Clock

Image: Texas Instruments
MCU Power Operating Modes

- 5 Low-Power Modes in total
  - Active Mode: CPU active, all clocks active
  - LPM0: CPU disabled, SMCLK and ACLK active
  - LPM3: CPU disabled, ACLK active
  - LPM4: CPU, all clocks disabled

Image: Texas Instruments
System Clock Summary

- MCU components are driven by MCLK, SMCLK, ACLK
  - Changing one input can affect multiple outputs
  - Changing one output can affect multiple components

- The relation between the three are limited by hardware
  - Signals are multiplied and/or divided

- Peripheral components can have different requirements
  - E.g., high-speed timer vs. low speed communication

- Choosing the clock source wisely can reduce power consumption significantly
Timers

Image: Texas Instruments
Timer Module

- 16-bit counter connected to either SMCLK or ACLK
  - Counts number of clock cycles
    - Measure elapsed time
    - Timestamp events
  - Set alarms at specific points
  - Continuous/Up-down mode
    - Overflow

- Capture/Compare Module
  - Read counter when an event occurs
  - Generate event when counter reaches specific value
Watchdog Timer

- Similar to Timer Module, except
  - Microcontroller resets when alarm is triggered
    - Clear counter periodically to avoid it (i.e. ‘feed the watchdog’)
    - Useful to reset microcontroller to a known state

- The watchdog is enabled by default after reset
Analog Signals

- **12-bit Analog-to-Digital Converter (ADC)**
  - Measure input voltage as a fraction of the reference voltage
    \[ N_{\text{ADC}} = 4095 \times \frac{V_{\text{in}} - V_{R-}}{V_{R+} - V_{R-}} \]
  - Analog sensors output voltage

- **Comparator**
  - Compare voltage between two analog signals

- **Digital-to-Analog Converter**
  - Outputs voltage
External Components

- PCB Antenna
- SMA Coax
- Humidity Temperature Sensor
- PAR Sensor
- TSR Sensor
- JTAG 8-pin 2mm IDC header
- Power I/O
- ADC[4]
- ADC[5]
- UART[1]
- UART[0]
- I2C[0]
- ADC[0-3.6-7]
- GPIO
- Reset
- User
- SVSin
- SVSout
- RX/TX RTS/DTR
- USB 2.0 UART/RS232 Functionality
- ST Flash 1024k (2.7V)
- Silicon Serial ID 1-wire
- SPI I/O
- TI MSP430 Microcontroller
- 10-pin + 6-pin IDC header
- SVS 2-pin IDC header
- JTAG
- Write Protection
- 4, 2, 2, 4
General Purpose Input/Output Pins (GPIO)

- **Input Pin**
  - High voltage reads ‘1’ – low voltage reads ‘0’
    - High – MCU supply voltage minus $\Delta$
    - Low – 0V plus $\Delta$

- **Output Pin**
  - Write ‘1’ sets voltage high – write ‘0’ sets voltage low

- **Interrupt capable Pin**
  - Interrupt program flow when input value changes
    - E.g. press button to make voltage go low
MCU to External Chip Communication

- GPIO pin
  - Control signals:
    - Turn peripherals on/off
    - Configure peripherals
  
- Data streams:
  - Manually transmit 1 bit at a time – ‘bit banging’
  - Inefficient, keeps CPU in Active Mode
1-Wire Bus (Timer Module)

- 1 signal line, multiple devices (master/slaves)
  - Designed for parasitic power
  - 64-bit unique serial number/address
  - Master pulls wire low to transmit
    - Duration determines whether it is a ‘1’ or ‘0’

Universal Synchronous/Asynchronous Receive/Transmit Module (USART)

- UART mode
  - Universal asynchronous receiver/transmitter (UART)
  - Transfers 1 ASCII character (7 or 8 bit) at a time
  - Self clocked
  - Receiver oversamples

- Very popular for interfacing with switches and routers

Universal Synchronous/Asynchronous Receive/Transmit Module (USART)

- Inter-Integrated Circuit (I2C) Mode
  - 2 signal lines, multiple devices (masters/slaves)
  - SCL – clock line (for timing)
  - SDA – data line
    - Speed: 400 kbit/s (although 3.4 Mbit/s standard exists)
  - Value is read/written on clock boundaries
    - 7-bit or 10-bit address space (1-bit used to signal read or write)
Universal Synchronous/Asynchronous Receive/Transmit Module (USART)

- **Serial Peripheral Interface Bus (SPI) Mode**
  - 3+1 signal lines, multiple devices (master/slaves)
    - Clock
    - MISO – master in, slave out
    - MOSI – master out, slave in
    - Slave select (enable slave when line is low)
  - Value is read/written on clock boundaries
  - Full duplex
Internal MCU Communication
Memory Map

- Shared Memory Space
- Memory Mapped I/O
  - Peripheral modules are accessed as regular variables (pointers)
  - Flash read, extension of RAM
  - Flash write, ‘peripheral module’

- Pros and cons
  - Global variables, no overhead
  - No memory protection

Images: Doina Bucur
Recap: GPIO

- 8 pins on one port
  - one bit per pin = 1 byte per port
  - E.g. set Port 1 pin 3 high:
    - P1OUT |= 0x04; // set value before direction to avoid outputting old value
    - P1DIR |= 0x04; // high means pin is output

<table>
<thead>
<tr>
<th>Port</th>
<th>Register</th>
<th>Short Form</th>
<th>Address</th>
<th>Register Type</th>
<th>Initial State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Input</td>
<td>P1IN</td>
<td>020h</td>
<td>Read only</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Output</td>
<td>P1OUT</td>
<td>021h</td>
<td>Read/write</td>
<td>Unchanged</td>
</tr>
<tr>
<td></td>
<td>Direction</td>
<td>P1DIR</td>
<td>022h</td>
<td>Read/write</td>
<td>Reset with PUC</td>
</tr>
<tr>
<td></td>
<td>Interrupt Flag</td>
<td>P1IFG</td>
<td>023h</td>
<td>Read/write</td>
<td>Reset with PUC</td>
</tr>
<tr>
<td></td>
<td>Interrupt Edge Select</td>
<td>P1IES</td>
<td>024h</td>
<td>Read/write</td>
<td>Unchanged</td>
</tr>
<tr>
<td></td>
<td>Interrupt Enable</td>
<td>P1IE</td>
<td>025h</td>
<td>Read/write</td>
<td>Reset with PUC</td>
</tr>
<tr>
<td></td>
<td>Port Select</td>
<td>P1SEL</td>
<td>026h</td>
<td>Read/write</td>
<td>Reset with PUC</td>
</tr>
</tbody>
</table>

Source: Texas Instruments
Recap: Timer Module

TACTL:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7-6</td>
<td>Input divider (IDx)</td>
</tr>
<tr>
<td>5-4</td>
<td>Mode control (MCx)</td>
</tr>
<tr>
<td>3</td>
<td>Unused</td>
</tr>
<tr>
<td>2</td>
<td>TACLRR</td>
</tr>
<tr>
<td>1</td>
<td>TAIE</td>
</tr>
<tr>
<td>0</td>
<td>TAIFG</td>
</tr>
</tbody>
</table>

**IDx**

- Bits: 00 /1
- Bits: 01 /2
- Bits: 10 /4
- Bits: 11 /8

**MCx**

- Bits: 00 Stop mode: the timer is halted
- Bits: 01 Up mode: the timer counts up to TACCR0
- Bits: 10 Continuous mode: the timer counts up to 0FFFFFFh
- Bits: 11 Up/down mode: the timer counts up to TACCR0 then down to 00000h

Start free-running timer:

- TACTL = (0x02 << 4);
Recap: Timer Module

- **Register**
  - **Timer_A control**
    - **Short Form**: TACTL
    - **Register Type**: Read/write
    - **Address**: 0160h
    - **Initial State**: Reset with POR
  - **Timer_A counter**
    - **Short Form**: TAR
    - **Register Type**: Read/write
    - **Address**: 0170h
    - **Initial State**: Reset with POR
  - **Timer_A capture/compare control 0**
    - **Short Form**: TACCTL0
    - **Register Type**: Read/write
    - **Address**: 0162h
    - **Initial State**: Reset with POR
  - **Timer_A capture/compare 0**
    - **Short Form**: TACCR0
    - **Register Type**: Read/write
    - **Address**: 0172h
    - **Initial State**: Reset with POR
  - **Timer_A capture/compare control 1**
    - **Short Form**: TACCTL1
    - **Register Type**: Read/write
    - **Address**: 0164h
    - **Initial State**: Reset with POR
  - **Timer_A capture/compare 1**
    - **Short Form**: TACCR1
    - **Register Type**: Read/write
    - **Address**: 0174h
    - **Initial State**: Reset with POR
  - **Timer_A capture/compare control 2**
    - **Short Form**: TACCTL2
    - **Register Type**: Read/write
    - **Address**: 0166h
    - **Initial State**: Reset with POR
  - **Timer_A capture/compare 2**
    - **Short Form**: TACCR2
    - **Register Type**: Read/write
    - **Address**: 0176h
    - **Initial State**: Reset with POR
  - **Timer_A interrupt vector**
    - **Short Form**: TAIV
    - **Register Type**: Read only
    - **Address**: 012Eh
    - **Initial State**: Reset with POR

- **Read free-running timer:**
  - **TACTL = (0x02 << 4); // start timer**
  - **localtime = TAR; // read value and store in ‘localtime’**
Missing Hardware Components

- What components are missing from the MCU?

- Floating Point Unit
  - Only found in high-powered systems
  - Software floating point: time and space overhead
Telosb Mote

- **Microcontroller**
  - Texas Instruments MSP430F1611

- **Peripherals**
  - 2x light sensors – ADC
  - Button – interrupt capable GPIO
  - Unique ID – 1-Wire
  - Temperature/humidity – I²C
  - External Flash – SPI
  - Radio – SPI, int. GPIO
  - USB – FTDI – UART
  - LEDs – GPIO
Schedule

- Week 1: Introduction and Hardware
- Week 2: Embedded Programming
- Week 3: Medium Access Control
- Week 4: Link Estimation and Tree Routing
- Week 5: IP Networking
- Week 6: Near Field Communication
- Week 7: (seminar, no lecture)
- Week 8: Energy Management
- Week 9: Review and Midterm
- Week 10: Time Synchronization
- Week 11: Localization
- Week 12: Energy Harvesting
- Week 13: (seminar, no lecture)
- Week 14: TBD