Day 02-1

Assignment:

- Hw 01, Due Friday
- Hw 02, Due Monday
- Hw 03, Due Next Thursday
-

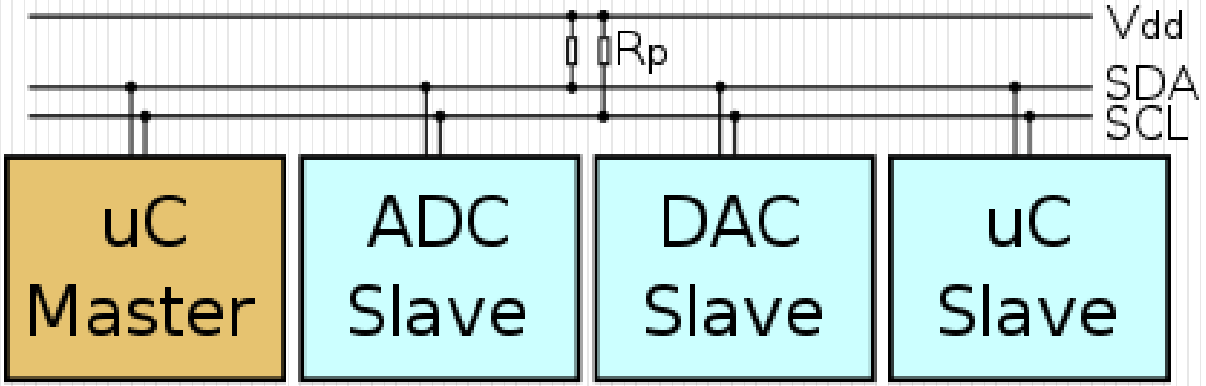
Today's Topics:

- Project Ideas
- I2C
- LED Matrix

http://www.gartner.com/newsroom/id/3114217?imm_mid=0d7750&cmp=em-iot-na-na-newsltr_20150827

02-2 I2C

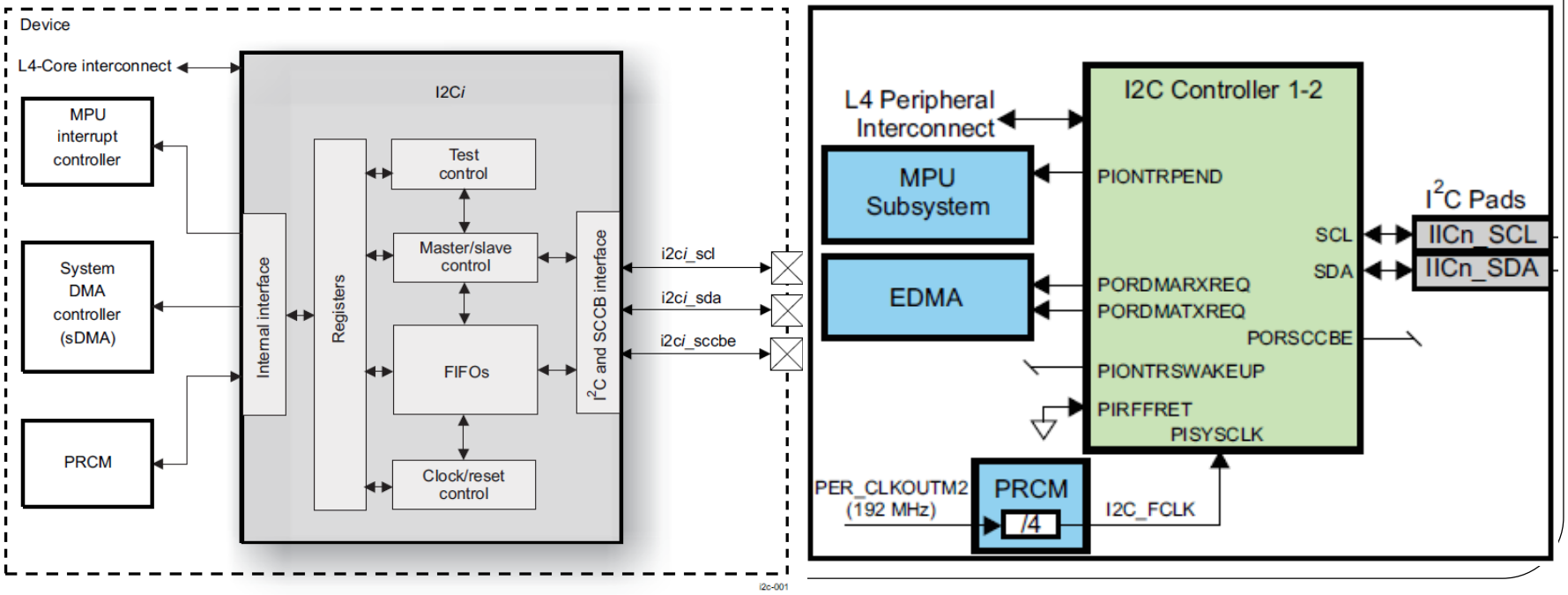
Interfacing with sensors over a serial bus



I²C

- "two-wire interface" standard
- Used to attach low-speed peripherals to embedded systems
- The Bone has three I²C controllers (Section 21 of TRM)

Figure 17-1. HS I²C Controllers Overview Block Diagram



Hardware - Bone

- You can see which ones are configured at boot time

```
bone$ dmesg –H | grep i2c
[  +0.001817] omap_i2c 44e0b000.i2c: could not find pctldev for node
/ocp/l4_wkup@44c00000/scm@210000/pinmux@800/pinmux_i2c0_pins, deferring
probe
[  +0.001349] omap_i2c 4802a000.i2c: bus 1 rev0.11 at 100 kHz
[  +0.002590] omap_i2c 4819c000.i2c: bus 2 rev0.11 at 100 kHz
[  +0.002904] i2c /dev entries driver
[  +0.000219] omap_i2c 44e0b000.i2c: bus 0 rev0.11 at 400 kHz
```

Three buses two running at 100 kHx and one at 400 kHz

Human readable

dmesg -Hw

Wait for more input

```
[  +0.147158] IPv6: ADDRCONF(NETDEV_CHANGE): SoftAp0: link becomes ready
[Sep 8 08:20] fbtft: module is from the staging directory, the quality is unknown, you have been warned.
[  +0.016886] fbtft_device: module is from the staging directory, the quality is unknown, you have been warned.
[  +0.001700] spidev spi1.0: spidev spi1.0 24000kHz 8 bits mode=0x00
[  +0.000026] spidev spi1.1: spidev spi1.1 24000kHz 8 bits mode=0x00
[  +0.000188] spidev spi1.0: Deleting spi1.0
[  +0.009426] fbtft_device: GPIOS used by 'adafruit28':
[  +0.000026] fbtft_device: 'reset' = GPIO113
[  +0.000008] fbtft_device: 'dc' = GPIO116
[  +0.000022] spidev spi1.1: spidev spi1.1 24000kHz 8 bits mode=0x00
[  +0.000014] spi spi1.0: fb_ili9341 spi1.0 32000kHz 8 bits mode=0x00
[  +0.043846] fb_ili9341: module is from the staging directory, the quality is unknown, you have been warned.
[  +0.344838] Console: switching to colour frame buffer device 40x30
[  +0.000694] graphics fb0: fb_ili9341 frame buffer, 320x240, 150 KiB video memory, 16 KiB DMA buffer memory, fps=20, spi1.0 at 32 MHz
[Sep 8 08:22] usb 1-1.4: USB disconnect, device number 8
[Sep 8 08:34] usb 1-1.1: USB disconnect, device number 3
[  +0.000030] usb 1-1.1.1: USB disconnect, device number 9
```

i2c - bone

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETN
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
I2C1_SCL	17	18	I2C1_SDA
I2C2_SCL	19	20	I2C2_SDA
I2C2_SCL	21	22	I2C2_SDA
GPIO_49	23	24	I2C1_SCL
GPIO_117	25	26	I2C1_SDA
GPIO_115	27	28	GPIO_123
GPIO_121	29	30	GPIO_122
GPIO_120	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AINO	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71

The first I2C bus is utilized for reading EEPROMS on cape add-on boards and can't be used for other digital I/O operations without interfering with that function, but you can still use it to add other I2C devices at available addresses.

The second I2C bus is available for you to configure and use.

Pin Mux - Clock

```
bone$ config-pin -i P9_21
Pin name: P9_21
Function if no cape loaded: gpio
Function if cape loaded: default gpio gpio_pu gpio_pd
gpio_input spi uart i2c pwm pru_uart
Function information: gpio0_3 default gpio0_3 gpio0_3
gpio0_3 gpio0_3 spi0_d0 uart2_txd i2c2_scl ehrpwm0b
pru_uart
Kernel GPIO id: 3
PRU GPIO id: 35
```

Pin Mux - Data

```
bone$ config-pin -i P9_22
Pin name: P9_22
Function if no cape loaded: gpio
Function if cape loaded: default gpio gpio_pu gpio_pd
gpio_input spi_sclk uart i2c pwm pru_uart
Function information: gpio0_2 default gpio0_2 gpio0_2
gpio0_2 gpio0_2 spi0_sclk uart2_rxd i2c2_sda ehrpwm0a
pru_uart
Kernel GPIO id: 2
PRU GPIO id: 34
```

Pin Mux - Warning

```
bone$ config-pin P9_21 i2c
bone$ config-pin P9_22 i2c
```

- Will make the i2c bus stop working

```
bone$ config-pin P9_21 gpio
bone$ config-pin P9_22 gpio
```

- Will fix it

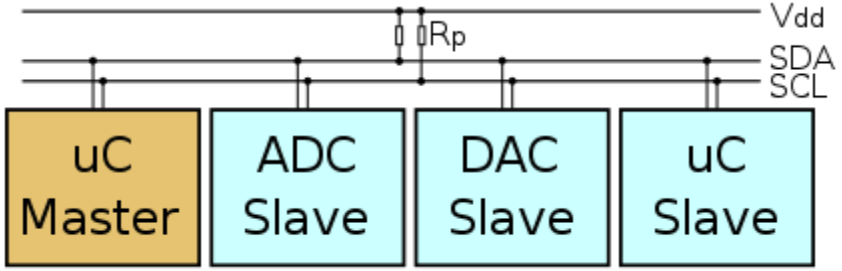
Hardware – TMP101

- Goal: Interface to a TMP101 temp sensor

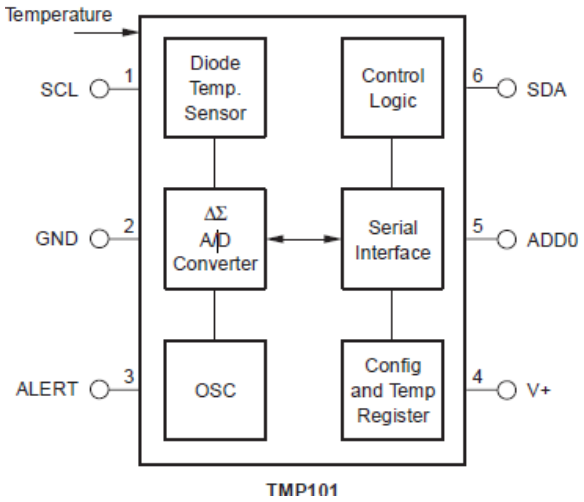
Parameter Name	Value
Typical Accuracy (°)	±2.0°C from −25°C to +85°C (max) ±3.0°C from −55°C to +125°C (max)
Supply Current (µA)	45µA, 0.1µA Standby
Resolution	9- to 12-bits,
Operating Voltage Range (V)	2.7V to 5.5V
Device Description	Serial Output Temp Sensor

http://www.ti.com/lit/gpn/tmp101

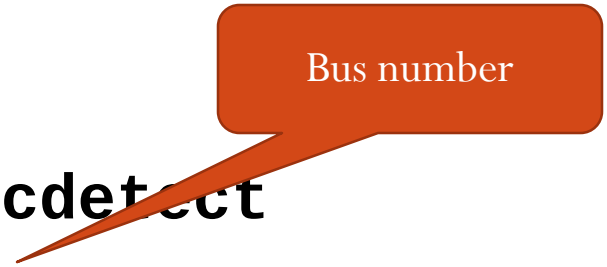
2-wire bus



- The two wires are
  - Serial Clock (SCL), is an input to the TMP101 and is used to clock data into and out of the TMP101.
  - Serial Data (SDA), is bidirectional and carries the data to and from the TMP101.
- The only other two pins on the TMP101 that you need to use are the Power Supply (Vdd) and Ground.



Software - bone



- See what's on a bus with **`i2cdetect`**

```
bone$ i2cdetect -y -r 2
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- 48 49 -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: 70 -- -- -- -- -- -- --
```

I have 2, TMP101's and an LED matrix.

- The TMP101's are at **`1001 000`** and **`1001 001`**
- Convert to hex **`0x48`** and **`0x49`**

Registers

- Each TMP101 has four registers

Table 2. Pointer Addresses of the TMP100 and TMP101 Registers

P1	P0	REGISTER
0	0	Temperature Register (READ Only)
0	1	Configuration Register (READ/WRITE)
1	0	T_{LOW} Register (READ/WRITE)
1	1	T_{HIGH} Register (READ/WRITE)

- Read with **`$ i2cget -y 2 0x48 00`**
- **`0x18`** which is 24C or 75.2F

Table 6. Configuration Register Format

BYTE	D7	D6	D5	D4	D3	D2	D1	D0
1	OS/ALERT	R1	R0	F1	F0	POL	TM	SD

Registers

Table 2. Pointer Addresses of the TMP100 and TMP101 Registers

P1	P0	REGISTER
0	0	Temperature Register (READ Only)
0	1	Configuration Register (READ/WRITE)
1	0	T_{LOW} Register (READ/WRITE)
1	1	T_{HIGH} Register (READ/WRITE)

- Read with **`$ i2cget -y 2 0x48 01`**
- **`0x80`** which is **`1000 0000`**

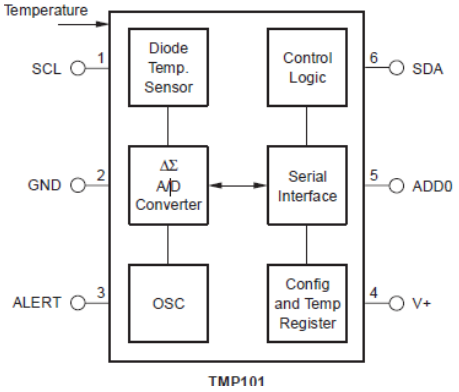


Table 6. Configuration Register Format

BYTE	D7	D6	D5	D4	D3	D2	D1	D0
1	OS/ALERT	R1	R0	F1	F0	POL	TM	SD

SD – Shutdown Mode
TM - Thermostat Mode
POL-Polarity
F1/F0 – Fault Queue
R1/R0 – Converter Resolution
OS – OS/Alert

Table 8. Resolution of the TMP100 and TMP101

R1	R0	RESOLUTION	CONVERSION TIME (typical)
0	0	9 Bits (0.5°C)	40ms
0	1	10 Bits (0.25°C)	80ms
1	0	11 Bits (0.125°C)	160ms
1	1	12 Bits (0.0625°C)	320ms

Read TMP101

```
#!/usr/bin/env python3
# Read a TMP101 sensor
# sudo apt install python3-smbus

import smbus
import time

bus = smbus.SMBus(2)
address = 0x49

while True:
    temp = bus.read_byte_data(address, 0)
    print (temp, end="\r")
    time.sleep(0.25)
```

Must install

Bus Number

Address of device

Register to read

exercises/displays/matrix8x8

- smbus commands

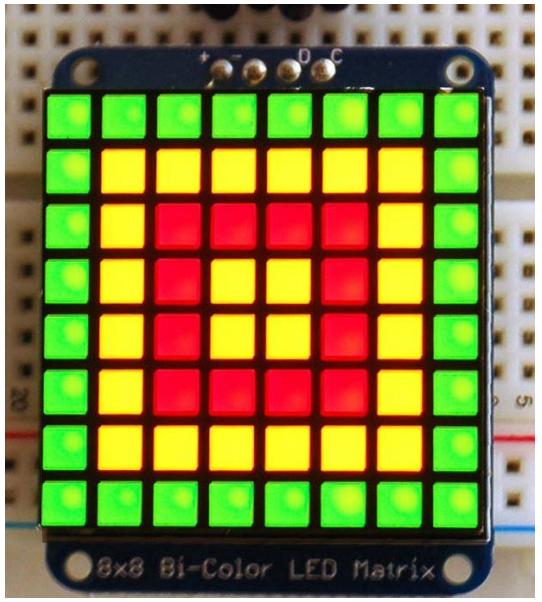
```
read_byte_data(int addr,char cmd)
write_byte_data(int addr,char cmd, char val)
write_byte_data(int addr,char cmd, char val)
write_i2c_block_data(int addr, char cmd, long vals[])
```

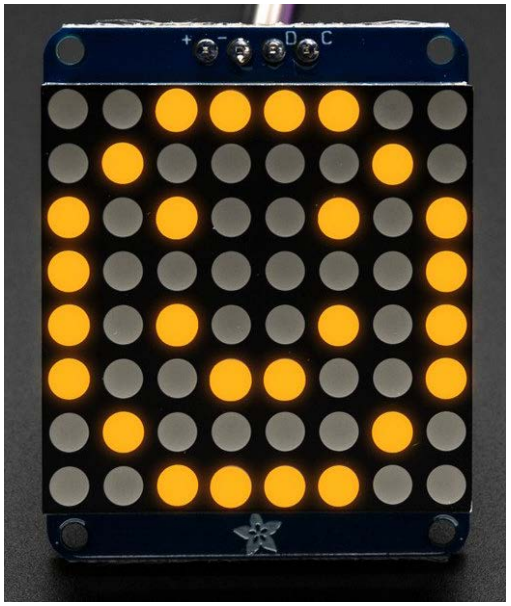
http://www.raspberry-projects.com/pi/programming-in-python/i2c-programming-in-python/using-the-i2c-interface-2

```
sudo apt install python3-smbus
```

LED Matrix

- In the lab you will be interfacing an I2C LED matrix
- Some are bicolor (red and green)
- Some are single color
- Both interface the same way: i2c
- How many wires to you need to control 128 LEDs?





https://www.adafruit.com/products/902

https://www.adafruit.com/products/871

exercises/displays/matrix8x8/i2cmatrix.py

```
#!/usr/bin/env python3
import smbus
import time
bus = smbus.SMBus(1)  # Use i2c bus 1
matrix = 0x70         # Use address


# The first byte is GREEN, the second is RED.
smile = [0x00, 0x3c, 0x00, 0x42, 0x28, 0x89, 0x04, 0x85,
    0x04, 0x85, 0x28, 0x89, 0x00, 0x42, 0x00, 0x3c
]
frown = [0x3c, 0x00, 0x42, 0x00, 0x85, 0x20, 0x89, 0x00,
    0x89, 0x00, 0x85, 0x20, 0x42, 0x00, 0x3c, 0x00
]
neutral = [0x3c, 0x3c, 0x42, 0x42, 0xa9, 0xa9, 0x89, 0x89,
    0x89, 0x89, 0xa9, 0xa9, 0x42, 0x42, 0x3c, 0x3c
]
```

exercises/displays/matrix8x8/i2cmatrix.py

```
# Start oscillator (p10)
bus.write_byte_data(matrix, 0x21, 0)
# Disp on, blink off (p11)
bus.write_byte_data(matrix, 0x81, 0)
# Full brightness (page 15)
bus.write_byte_data(matrix, 0xe7, 0)
```

exercises/displays/matrix8x8/i2c

```
bus.write_i2c_block_data(matrix, 0, frown)
time.sleep(delay)

bus.write_i2c_block_data(matrix, 0, neutral)

for fade in range(0xe0, 0xef, 1):
    bus.write_byte_data(matrix, fade, 0)
    time.sleep(delay/10)

bus.write_i2c_block_data(matrix, 0, smile)
```

Writing one column

```
bus.write_i2c_block_data(matrix,
                         0x04, [0xff]);
```

- Turns on the third column of *green* LEDs, without writing the other columns
- Try it!

LED Matrix

- Goal: Etch-a-Sketch on the LED Matrix!