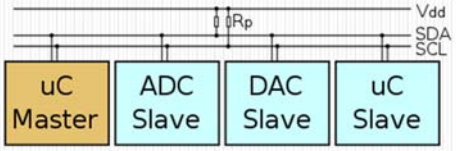
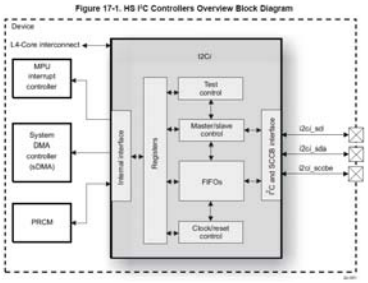
# 02-2 I2C

Interfacing with sensors over a serial bus




## I²C

- "two-wire interface" standard
- Used to attach low-speed peripherals to embedded systems
- Beagle has four I²C controllers (Section 17 of TRM)




## Hardware

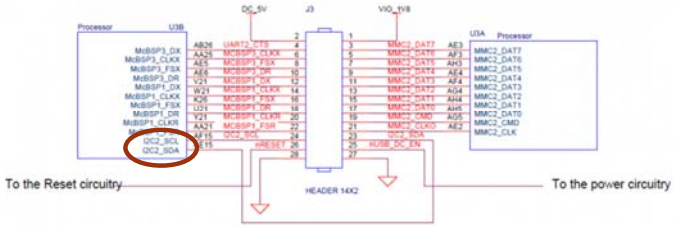
- You can see which ones are configured at boot time

```
dmesg | grep i2c
[    0.000000] Beagle cameraboard: registering i2c2 bus for lbcm3m1
[    9.123779] i2c_omap i2c_omap.1: bus 1 rev4.0 at 2600 kHz
[    9.132568] i2c_omap i2c_omap.2: bus 2 rev4.0 at 400 kHz
[    9.132904] i2c_omap i2c_omap.3: bus 3 rev4.0 at 100 kHz
[   10.476806] input: twl4030_pwrbutton as
/devices/platform/i2c_omap.1/i2c-1/1-
0049/twl4030_pwrbutton/input/input1
[   10.487579] i2c /dev entries driver
```

- Three buses each running at a difference speed

## Bus 2

- Bus 2 is brought out on the expansion header




- These signals are 1.8V

## Hardware – TC74

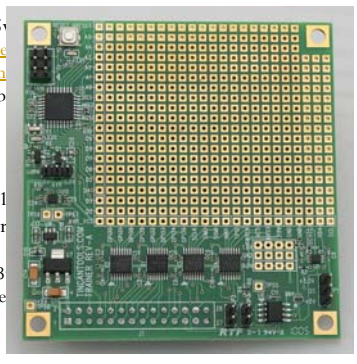
- Goal: Interface to a TC74 temp sensor

| Parameter Name | Value |
|---|---|
| Typical Accuracy (°) | 0.5 |
| Max Input/ Supply Current (µA) | 350 |
| Max. Accuracy @ 25° (°) | 2 |
| Temp. Range (°C) | -40 to +125 |
| Operating Voltage Range (V) | 2.7 to 5.5 |
| Device Description | Serial Output Temp Sensor |

http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010749#1

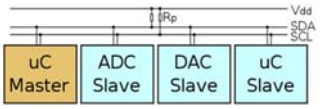
## BeagleBoard Trainer

- http://elinux.org/BeagleBoard_Trainer
- Trainer Features:
- I2C interface(+3.3v or +5v
  - Can be used with the Ninte
  - Pin compatible with WiiCh
  - WiiChuk Adapter is availab
  - Wii Nunchuk project
- SPI inteface (+3.3v)
- GPIO's(+3.3v)
- Large prototyping area (0.1
- Atmega328 processor (user
  - Arduino compatible
  - ATmega328 power is +3.3
  - Communicate to the ATme



## 2-wire bus

- The two wires are
  - Serial Clock (SCLK on the data sheet, SCL on the Beagle), is an input to the TC74 and is used to clock data into and out of the TC74.
  - Serial Data (SDA on both), is bidirection and carries the data to and from the TC74.
- The only other two pins on the TC74 that you need to use are the Power Supply (Vdd) and Ground.




## Software

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

```
# i2cdetect -y -r 2
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- UU -- -- --
40: -- -- -- -- -- -- -- -- 48 -- 4a -- -- -- -- --
50: 50 -- -- 53 -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

- I have 2 TC74's and a 3-axis accelerometer.
- The TC74's are at **`1001 000`** and **`1001 010`**
- Convert to hex **`0x48`** and **`0x4a`**

## Registers

- Each TC74 has two registers

| Command | Code | Function |
|---|---|---|
| RTR | 0x00 | Read Temperature (TEMP) |
| RWCR | 0x01 | Read/Write Configuration (CONFIG} |

- Read with **`$ i2get -y 2 0x48 0`**
- **`0x1c`** which is 28C or 82.4F, rather warm

## I²C via C

```
int main(int argc, char *argv[]) {
        char *end;
        int res, i2cbus, address, size, file;
        int daddress;
        char filename[20];

        /* handle (optional) flags first */
        if(argc < 3) {
                fprintf(stderr,
                        "Usage:  %s <i2c-bus> <i2c-address> <register>\n",
                        argv[0]);
                exit(1);
        }
        i2cbus   = atoi(argv[1]);
        address  = atoi(argv[2]);
        daddress = atoi(argv[3]);
        size = I2C_SMBUS_BYTE;
```

## I²C via C

```
    sprintf(filename, "/dev/i2c-%d", i2cbus);
    file = open(filename, O_RDWR);
    if (file < 0) {
        if (errno == ENOENT) {
            fprintf(stderr, "Error: Could not open file "
            "/dev/i2c-%d: %s\n", i2cbus, strerror(ENOENT));
        } else {
            fprintf(stderr, "Error: Could not open file "
                "`%s': %s\n", filename, strerror(errno));
            if (errno == EACCES)
                fprintf(stderr, "Run as root?\n");
        }
        exit(1);
    }
```

## I²C via C

```
    if (ioctl(file, I2C_SLAVE, address) < 0) {
        fprintf(stderr,
                "Error: Could not set address to 0x%02x: %s\n",
                address, strerror(errno));
        return -errno;
    }

    res = i2c_smbus_write_byte(file, daddress);
    if (res < 0) {
        fprintf(stderr, "Warning - write failed, filename=%s,
                daddress=%d\n", filename, daddress);
    }
    res = i2c_smbus_read_byte_data(file, daddress);
    close(file);
```