Aakash Project Report

Embedded Temperature Interface using Aakash and Raspberry Pi

Project by:

Bestin Johnson (108112020)
C Ganesh Sundar (107112017)

Acknowledgement

We are extremely grateful to Dr. Venkataramani and Dr. Balasundaram for providing this opportunity.

We also thank Mrs. Malathi and all the OCTA members for guiding us through this internship.

What have we done?

Constructed an embedded interface for measuring and displaying temperature on the Aakash Tablet using Raspberry Pi

How did we do?

Components used:

==> LM35 Temperature Sensor

==> Analog output to ADC

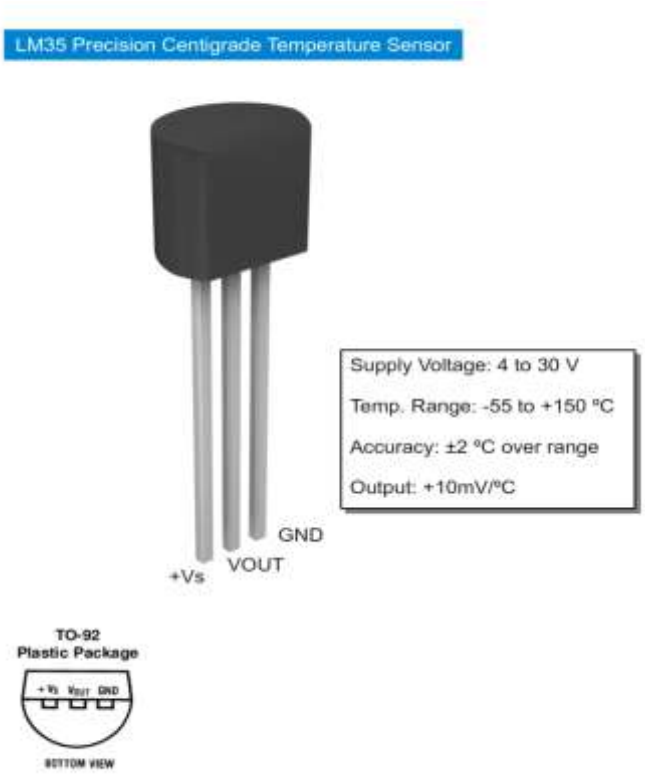
==> MCP3008 (ADC + parallel in serial out shift register)

==> Raspberry Pi

==> USB interface

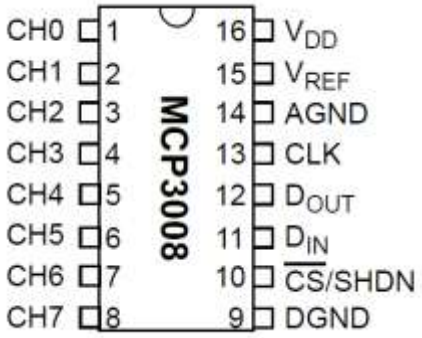
==> Aakash Tablet access via SSH

LM35 - Temperature Sensor



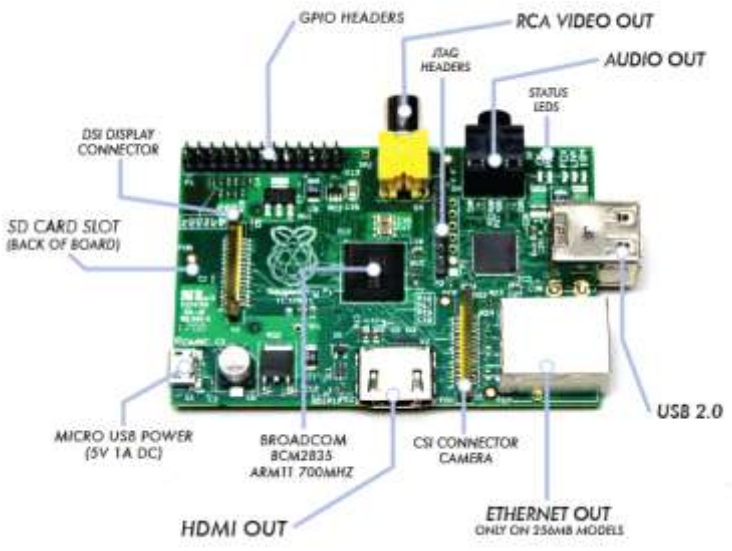
MCP3008

The MCP3008 is a 10bit 8-channel Analogue-to-digital converter (ADC) with SPI interface.



Raspberry Pi

- Credit-card-sized single-board computer
- Broadcom BCM2835 system on a chip (SoC)
- an ARM1176JZF-S 700 MHz processor
- VideoCore IV GPU
- 512 megabytes of RAM
- SD card for booting and persistent storage
- USB ports
- Ethernet controller



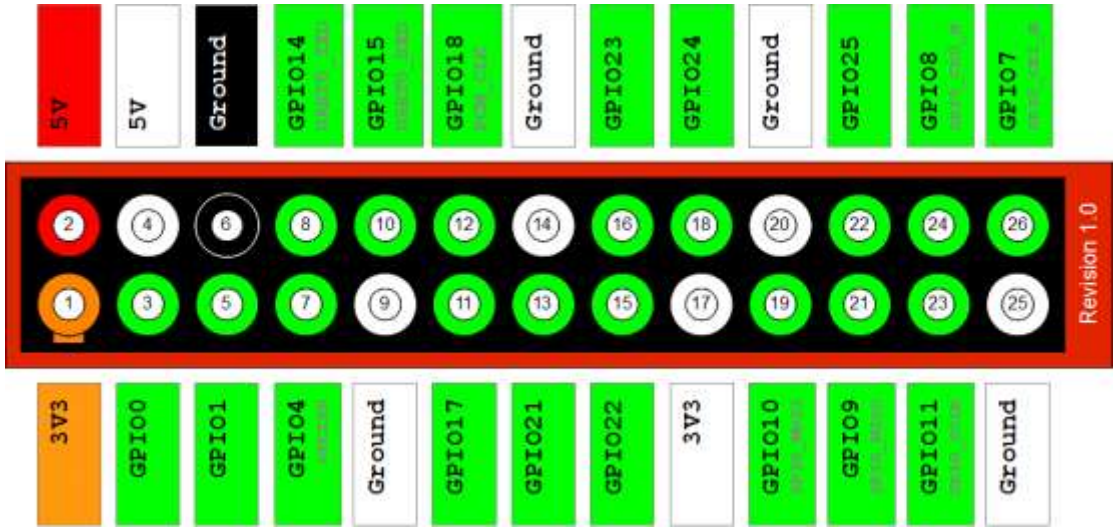


fig: Raspberry Pi pinout

Program code on Raspberry Pi

The script is fairly simple. Half of the code (the readadc function) is a function that will 'talk' to the MCP3008 chip using four digital pins to 'bit bang' the SPI interface.The MCP3008 is a 10-bit ADC. That means it will read a value from 0 to 1023 (2^^10 = 1024 values) where 0 is the same as 'ground' and '1023' is the same as '3.3 volts'

Every 5 seconds the following process are done:
1.Read the adc value on channel 0 (temperature sensor).
2.Convert the adc value to millivolts: millivolts = read_adc0 * (3300.0 / 1023.0).
3.Convert the millivolts value to a celsius temperature: temp_C = millivolts/10.
4.Convert the celsius temperature to a fahrenheit temperature: temp_F = (temp_C * 9.0 / 5.0) + 32).

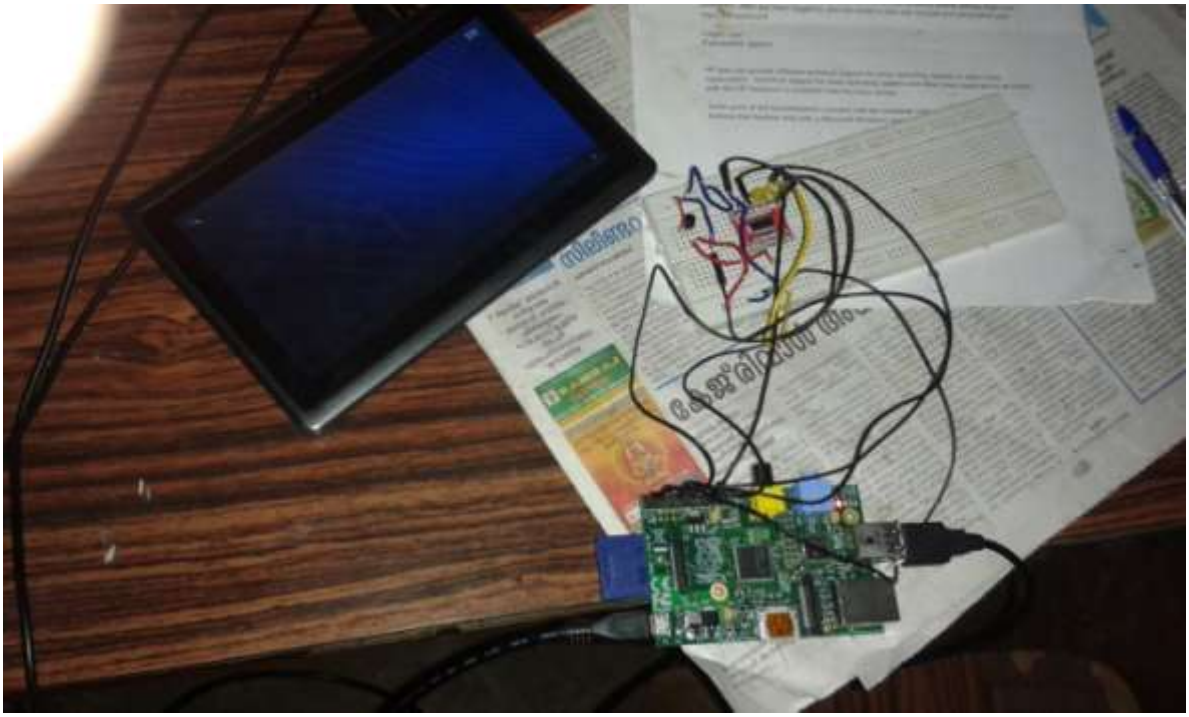
Python code:

```
#!/usr/bin/env python
import time
import os
```

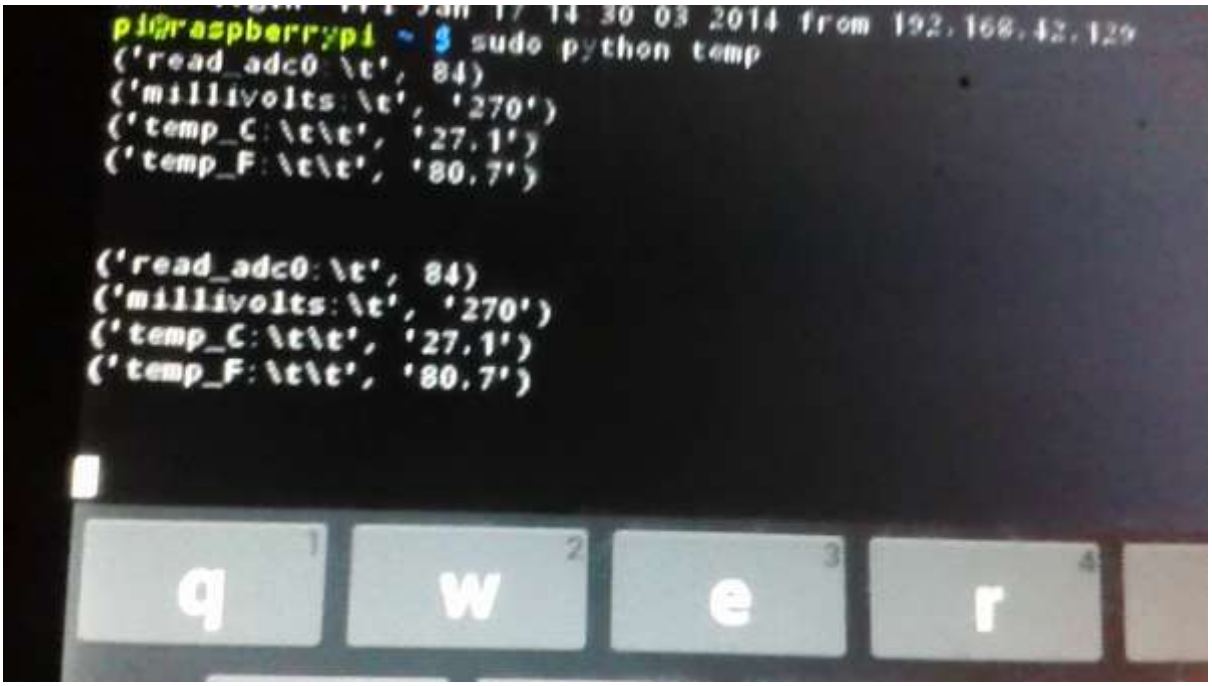
```
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
DEBUG = 1

# read SPI data from MCP3008 chip, 8 possible adc's (0 thru 7)
def readadc(adcnum, clockpin, mosipin, misopin, cspin):
        if ((adcnum > 7) or (adcnum < 0)):
                return -1
        GPIO.output(cspin, True)

        GPIO.output(clockpin, False)  # start clock low
        GPIO.output(cspin, False)     # bring CS low

        commandout = adcnum
        commandout |= 0x18  # start bit + single-ended bit
        commandout <<= 3    # we only need to send 5 bits here
        for i in range(5):
                if (commandout & 0x80):
                        GPIO.output(mosipin, True)
                else:
                        GPIO.output(mosipin, False)
                commandout <<= 1
                GPIO.output(clockpin, True)
                GPIO.output(clockpin, False)

        adcout = 0
        # read in one empty bit, one null bit and 10 ADC bits
        for i in range(12):
                GPIO.output(clockpin, True)
                GPIO.output(clockpin, False)
                adcout <<= 1
                if (GPIO.input(misopin)):
                        adcout |= 0x1

        GPIO.output(cspin, True)

        adcout /= 2       # first bit is 'null' so drop it
        return adcout

# change these as desired - they're the pins connected from the
# SPI port on the ADC to the Cobbler
SPICLK = 18
SPIMISO = 23
SPIMOSI = 24
SPICS = 25

# set up the SPI interface pins
GPIO.setup(SPIMOSI, GPIO.OUT)
GPIO.setup(SPIMISO, GPIO.IN)
GPIO.setup(SPICLK, GPIO.OUT)
GPIO.setup(SPICS, GPIO.OUT)

# temperature sensor connected channel 0 of mcp3008
adcnum = 0

while True:
        # read the analog pin (temperature sensor LM35)
        read_adc0 = readadc(adcnum, SPICLK, SPIMOSI, SPIMISO, SPICS)

        # convert analog reading to millivolts = ADC * ( 3300 / 1024
)
        millivolts = read_adc0 * ( 3300.0 / 1024.0)
```

```
        # 10 mv per degree
        temp_C = ((millivolts - 100.0) / 10.0) - 40.0

        # convert celsius to fahrenheit
        temp_F = ( temp_C * 9.0 / 5.0 ) + 32

        # remove decimal point from millivolts
        millivolts = "%d" % millivolts

        # show only one decimal place for temprature and voltage
readings
        temp_C = "%.1f" % temp_C
        temp_F = "%.1f" % temp_F

        if DEBUG:
                print("read_adc0:\t", read_adc0)
                print("millivolts:\t", millivolts)
                print("temp_C:\t\t", temp_C)
                print("temp_F:\t\t", temp_F)
                print("\n")

        # hang out and do nothing for 5 seconds,
        time.sleep(5)
```

Pictures:



Interfacing Raspberry Pi and Aaakash Tab 1



Displaying temperature on Aakash 1

Thank You!