RFID BASED MESS COUPON SYSTEM

November 19, 2016

Due Date: 19 Nov'16

Group Members: Devishi Kesar (2015024)
Gaurav Gehlot (2015030)
Gunkirat Kaur (2015032)
Kushagra Arora (2015049)
Nikhil Hassija (2015065)

Instructor: Prof. Reddy

1. Concepts Involved

1. Interrupt Handling
2. Memory Addressing
3. Subroutines
4. Delays
5. Serial Communication

1.1. Interrupt Handling

As seen below in the diagram the RPi receives interrupts from the RFID reader ,the reader creates an interrupt and sends it to the RPi which is followed by reading of data,showing on LCD and updating on the server. An interrupt is like having an automatic postman detector that will tell you for sure when the postman arrives, so you can get on with something else. You now know you will not miss that knock on the door and end up with one of those "we tried to deliver your item but you were out and the collection office is closed for the next two days, so enjoy the wait" cards. Thus interrupts were important for the easy functioning of the program.

1.2. Memory Addressing

On a computer you write to a specified 'memory address'. This address is recognised by the system as a hardware address, and the appropriate hardware receives or sends the appropriate value.
Most hardware systems have many different registers that can be set or read. Some might have a few, some might have many. These registers will be grouped into a continuous range. A base pointer points to the first in the range, and you write to, for example, the second port with $base_pointer + 1$. You don't have to, you could write direct to a pointer, but using an offset makes things easier to work with.
The Raspberry Pi recognises a massive range of hardware registers at the address 0x20000000. A range of registers that control clock systems are accessed from $BCM2708_PERI_BASE + 0x101000$. The registers that control the I2S clock are the 38th and 39th register in that block, written to using $BCM2708_PERI_BASE + 0x101000 + 0x26$ and 0x27

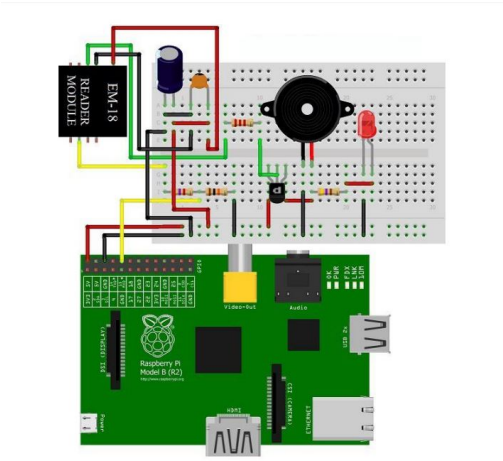

Figure 1: Connections in circuit

1.3. Subroutines

Subroutines are the user defined functions created to make understanding the code easier,make it more flexible and allow prototyping.The subroutines written in the python code make the program execution easier and allow the work to go as per the requirement.

1.4. Delays

Delay is an important concept in such a hardware implementation as they allow the processor to work on one task and save from hazards like "resource unavailable". Delays are inserted in between tasks to ensure that the task is finished according to the priority decided by us and given time to be observed if needed.

1.5. Serial Communication

We have used Serial Peripheral Interface(SPI) for the project.The Raspberry Pi is equipped with one SPI bus that has 2 chip selects.The SPI master driver is disabled by default on Raspbian.

2. The working

RFID (Radio Frequency Identification) uses electromagnetic fields to read, monitor and transfer data from tags attached to different objects. It is not necessary that the cards are to be in visibility of the reader, it can be embedded in the tracked object. The tags can be actively powered from a power source or can be passively powered form the incoming electromagnetic fields.
RFID is an electronics device which has two parts - one is RFID Reader and other is RFID tag or Card. When we put RFID tag near to the RFID reader, it reads tag data serially. RFID tag has 12 digit character code in a coil. This RFID is working at baud rate of 9600 bps. RFID uses electromagnet to transfer data from Reader to Tag or Tag to Reader. When a person put their RFID tag near over the RFID reader to scan, RFID reads tag's data and sends it to Raspberry Pi. Then Raspberry Pi reads the Unique Identification Number of that RFID tag and then compares this data with predefined data or information. If data is matched with predefined data, then Raspberry Pi increments the attendance of the tag's person by one and if matched is not matched then microcontroller shows 'Invalid Card' message on LCD. Here we have taken 2 RFID tags.

3. Source Code

```

import RPi.GPIO as GPIO
import MFRC522
import signal
import time
continue_reading = True
import urllib2

url = "http://192.168.57.55:8000/home/"

MIFAREReader = MFRC522.MFRC522()
GPIO.setwarnings(False)
try:
    while continue_reading:

        (status,TagType) = MIFAREReader.MFRC522_Request(MIFAREReader.PICC_REQIDL)

        if status == MIFAREReader.MI_OK:
            pass

        (status,uid) = MIFAREReader.MFRC522_Anticoll()

        if status == MIFAREReader.MI_OK:

            uid = str(uid[0])+","+str(uid[1])+","+str(uid[2])+","+str(uid[3])
            url += uid
#           print uid
            response = urllib2.urlopen(url)
            data = response.read()
            print data
            continue_reading = False
            GPIO.cleanup()
except :
    print "exiting"
```

```
#include <wiringPi.h>
#include <lcd.h>
//USE WIRINGPI PIN NUMBERS
#define LCD_RS  25               //Register select pin
#define LCD_E   24               //Enable Pin
#define LCD_D4  23               //Data pin 4
#define LCD_D5  22               //Data pin 5
#define LCD_D6  21               //Data pin 6
#define LCD_D7  29               //Data pin 7
```

```

#include <stdio.h>
#include <stdlib.h> // For exit() function

int main()
{

    char s[1000];
    int c,i;
    i = 0;
    FILE *fptr;
    fptr = fopen("file1.txt", "r");
    if(!fptr)
    return 0;

    fscanf(fptr, "%[^\n]", s);
        fclose(fptr);


//   char c[100] = "abc";

// printf(s);
 int lcd;
    wiringPiSetup();
    lcd = lcdInit (2, 16, 4, LCD_RS, LCD_E, LCD_D4, LCD_D5, LCD_D6, LCD_D7, 0, 0, 0, ↩
        0);
 lcdClear(lcd);
    lcdPuts(lcd, s);
    sleep(3);
    lcdClear(lcd);
    lcdPuts(lcd, "card read");
    return 0;
}
```

```
# /bin/bash

gcc -o lcd fourbit.c -lwiringPi -lwiringPiDev
while true
do
    sudo python Read.py > file1.txt
    TEXT=‘cat file1.txt‘
    if [ "$TEXT" == "exiting" ]
    then
        break
    fi
    sudo ./lcd
```

```
done
```

```

lcd:     file format elf64-x86-64


Disassembly of section .text:

0000000000000000 <main>:

#include <stdio.h>
#include <stdlib.h> // For exit() function

int main()
{
   0:   55                      push   %rbp
   1:   48 89 e5                mov    %rsp,%rbp
   4:   48 81 ec 00 04 00 00    sub    $0x400,%rsp
   b:   64 48 8b 04 25 28 00    mov    %fs:0x28,%rax
  12:   00 00
  14:   48 89 45 f8             mov    %rax,-0x8(%rbp)
  18:   31 c0                   xor    %eax,%eax

    char s[1000];
    int c,i;
    i = 0;
  1a:   c7 85 00 fc ff ff 00    movl   $0x0,-0x400(%rbp)
  21:   00 00 00
    FILE *fptr;
    fptr = fopen("file1.txt", "r");
  24:   be 00 00 00 00          mov    $0x0,%esi
  29:   bf 00 00 00 00          mov    $0x0,%edi
  2e:   e8 00 00 00 00          callq  33 <main+0x33>
  33:   48 89 85 08 fc ff ff    mov    %rax,-0x3f8(%rbp)
    if(!fptr)
  3a:   48 83 bd 08 fc ff ff    cmpq   $0x0,-0x3f8(%rbp)
  41:   00
  42:   75 0a                   jne    4e <main+0x4e>
    return 0;
  44:   b8 00 00 00 00          mov    $0x0,%eax
  49:   e9 cc 00 00 00          jmpq   11a <main+0x11a>

    fscanf(fptr, "%[^\n]", s);
  4e:   48 8d 95 10 fc ff ff    lea    -0x3f0(%rbp),%rdx
  55:   48 8b 85 08 fc ff ff    mov    -0x3f8(%rbp),%rax
  5c:   be 00 00 00 00          mov    $0x0,%esi
```

```
  61:   48 89 c7                mov    %rax,%rdi
  64:   b8 00 00 00 00          mov    $0x0,%eax
  69:   e8 00 00 00 00          callq  6e <main+0x6e>
        fclose(fptr);
  6e:   48 8b 85 08 fc ff ff    mov    -0x3f8(%rbp),%rax
  75:   48 89 c7                mov    %rax,%rdi
  78:   e8 00 00 00 00          callq  7d <main+0x7d>

//   char c[100] = "abc";

// printf(s);
 int lcd;
    wiringPiSetup();
  7d:   e8 00 00 00 00          callq  82 <main+0x82>
    lcd = lcdInit (2, 16, 4, LCD_RS, LCD_E, LCD_D4, LCD_D5, LCD_D6, LCD_D7, 0, 0, 0, ↩
        0);
  82:   48 83 ec 08             sub    $0x8,%rsp
  86:   6a 00                   pushq  $0x0
  88:   6a 00                   pushq  $0x0
  8a:   6a 00                   pushq  $0x0
  8c:   6a 00                   pushq  $0x0
  8e:   6a 1d                   pushq  $0x1d
  90:   6a 15                   pushq  $0x15
  92:   6a 16                   pushq  $0x16
  94:   41 b9 17 00 00 00       mov    $0x17,%r9d
  9a:   41 b8 18 00 00 00       mov    $0x18,%r8d
  a0:   b9 19 00 00 00          mov    $0x19,%ecx
  a5:   ba 04 00 00 00          mov    $0x4,%edx
  aa:   be 10 00 00 00          mov    $0x10,%esi
  af:   bf 02 00 00 00          mov    $0x2,%edi
  b4:   e8 00 00 00 00          callq  b9 <main+0xb9>
  b9:   48 83 c4 40             add    $0x40,%rsp
  bd:   89 85 04 fc ff ff       mov    %eax,-0x3fc(%rbp)
 lcdClear(lcd);
  c3:   8b 85 04 fc ff ff       mov    -0x3fc(%rbp),%eax
  c9:   89 c7                   mov    %eax,%edi
  cb:   e8 00 00 00 00          callq  d0 <main+0xd0>
    lcdPuts(lcd, s);
  d0:   48 8d 95 10 fc ff ff    lea    -0x3f0(%rbp),%rdx
  d7:   8b 85 04 fc ff ff       mov    -0x3fc(%rbp),%eax
  dd:   48 89 d6                mov    %rdx,%rsi
  e0:   89 c7                   mov    %eax,%edi
  e2:   e8 00 00 00 00          callq  e7 <main+0xe7>
    sleep(3);
  e7:   bf 03 00 00 00          mov    $0x3,%edi
  ec:   b8 00 00 00 00          mov    $0x0,%eax
  f1:   e8 00 00 00 00          callq  f6 <main+0xf6>
```

```
 91      lcdClear(lcd);
 92    f6:   8b 85 04 fc ff ff       mov    -0x3fc(%rbp),%eax
 93    fc:   89 c7                   mov    %eax,%edi
 94    fe:   e8 00 00 00 00          callq  103 <main+0x103>
 95      lcdPuts(lcd, "card read");
 96   103:   8b 85 04 fc ff ff       mov    -0x3fc(%rbp),%eax
 97   109:   be 00 00 00 00          mov    $0x0,%esi
 98   10e:   89 c7                   mov    %eax,%edi
 99   110:   e8 00 00 00 00          callq  115 <main+0x115>
100      return 0;
101   115:   b8 00 00 00 00          mov    $0x0,%eax
102  }
103   11a:   48 8b 4d f8             mov    -0x8(%rbp),%rcx
104   11e:   64 48 33 0c 25 28 00    xor    %fs:0x28,%rcx
105   125:   00 00
106   127:   74 05                   je     12e <main+0x12e>
107   129:   e8 00 00 00 00          callq  12e <main+0x12e>
108   12e:   c9                      leaveq
109   12f:   c3                      retq
```

4. Scope

The project has various manifestations and can be sued for various purposes. The use of RFID is not limited to mess coupons it can further be used for attendance etc. The RFID system implemented by us for mess coupons allows us to easy the system of mess coupons , buying as well as using it.

5. Video Link

https://drive.google.com/file/d/0B1y71A2r6Brbd2p0T1BwaHZWVzA/view?usp=sharing