AC 2012-3456: EMBEDDED RF SYSTEM DESIGN WITH THE RFPIC12F675

Prof. Omer Farook, Purdue University, Calumet

Omer Farook is a member of the faculty of electrical and computer engineering technology at Purdue University, Calumet. Farook received the diploma of licentiate in mechanical engineering and B.S.M.E. in 1970 and 1972 respectively. He further received B.S.E.E. and M.S.E.E. degrees in 1978 and 1983, respectively, from Illinois Institute of Technology. Farook's current interests are in the areas of embedded system design, hardware software interfacing, digital communication, networking, image processing, and biometrics, C++, PHP, and Java Languages. He has a keen interest in pedagogy and instruction delivery methods related to distance learning.

Mr. Paul Lapsansky, Purdue University, Calumet

Paul Lapsansky graduated with a B.S. in E.E.T. from Purdue University, Calumet, and is currently working on completing his M.S. in technology. He is presently the Manager of Technical Services for Valparaiso University.

Prof. Chandra Sekhar, Purdue University, Calumet

Chandra R. Sekhar is a member of the faculty of the Electrical and Computer Engineering Technology Department at Purdue University, Calumet. Sekhar earned a bachelor's degree in chemistry from the University of Madras (India), a diploma in instrumentation from Madras Institute of Technology, and a master's Degree in electrical engineering from the University of Pennsylvania. Sekhar's primary teaching and research focus is in the areas of biomedical and process control instrumentation and clinical engineering.

Dr. Jai P. Agrawal, Purdue University, Calumet

Jai P. Agrawal is a professor in electrical and computer engineering technology at Purdue University, Calumet. He received his Ph.D. in electrical engineering from University of Illinois, Chicago, in 1991, dissertation in power electronics. He also received M.S. and B.S. degrees in electrical engineering from Indian Institute of Technology, Kanpur, India, in 1970 and 1968, respectively. His expertise includes analog and digital electronics design, power electronics, nanophotonics, and optical/wireless networking systems. He has designed several models of high frequency oscilloscopes and other electronic test and measuring instruments as an entrepreneur. He has delivered invited short courses in Penang, Malaysia and Singapore. He is also the author of a textbook in power electronics, published by Prentice-Hall, Inc. His professional career is equally divided in academia and industry. He has authored several research papers in IEEE journals and conferences. His current research is focused on renewable energy technology, smart energy grid.

Prof. Ashfaq Ahmed, Purdue University, Calumet

Ashfaq Ahmed is a professor of the Electrical and Computer Engineering Technology Department at Purdue University, Calumet. Ahmed received his bachelor's of science degree in electrical engineering from the University of Karachi in 1973 and master's of applied science degree in 1978 from University of Waterloo. He is the author of a textbook in power electronics, published by Prentice-Hall. He is a registered Professional Engineer in the state of Indiana. He is senior member of IEEE. Ahmed's current interests include embedded system design, C++, and VHDL design.

Embedded RF System Design with the rfPIC12F675

Abstract

The paper dwells on "Short range low power wireless technology" which is playing an ever increasing key role in embedded systems design freeing the system developer from the limitations and constraints of a wired connection. This paper describes how the Embedded RF System Design is incorporated in a senior / graduate level course in the Electrical and Computer Engineering Technology Program. The content of this course is offered both as a standalone course and also an abridged version is adopted as a Final course project in the Digital Communication course. The gist of this course can also be incorporated in a course of Robotics.

The course is based on embedded RF System Design utilizing Microchip Technology's rfPIC12F675 to provide data transmission between two embedded systems. The rfPIC12F675 is a low cost, high performance Microcontroller with UHF ASK / FSK short-range radio transmitter. The receiving end consists of a PIC16F676 (or any microcontroller) coupled with an rfRXD0420 UHF receiver that has been designed to complement the rfPIC12F675. The paper elaborates a general purpose radio controlled robot used in the course. Firmware development consists of a mix of both C and assembler and is centered around but not limited to Microchip MPLAB IDE, the MPASM assembler and the included HI-TECH C compiler. The rfPIC12F675 and the rfRXD0420, combined with a standard PIC microcontroller and the Microchip MPLAB development environment provides for an ideal development environment for introducing short-range RF to embedded systems design.

The paper further elaborates the pedagogy of the course delivery which is based on "Interactive and Collaborative Learning model". The course is conducted in a lab or studio like settings, that integrates both lecture and laboratory work in the same settings, with students working in teams.

I. Introduction

Integration of RF principles with that of embedded systems principles provide to the class room an added interest and the content area that provides relevance to content of the subject area. The fast changing field of radio frequency (RF) communication technology is one of the disciplines strongly emphasized within the electronics and computer engineering technology (ECET) programs. The approach taken by our institute is to integrate communication theory in conjunction with Embedded System classes. The material presented here is a link in this approach.

II. Hardware Design

The core of this course is the Microchip Technology's rfPIC12F675. The rfPIC12F675 is a FLASH-based microcontroller with an ASK/FSK transmitter. The rfPIC12F675 is essentially a PIC12F675 with the addition of a ASK/FSK transmitter and so includes the same features of the PIC12F675 including[1]:

- 1K FLASH memory
- 64 bytes of SRAM
- 128 bytes of EEPROM storage
- Six general purpose I/O pins
- Analog comparator
- 10-bit analog-to-digital converter

The rfPIC12F675 adds the following:

- ASK/FSK modulation
- Adjustable output power: -12 to +6dBm
- Crystal controlled Phase locked Loop
- 20-pin SSOP package

The rfPIC12F675 is available in three different frequency ranges (Table 1). For this project, the rfPIC12F675F utilizing ASK modulation was chosen although any of the three frequency ranges would have worked. Applications for the rfPIC12F675 include:

- Automotive Remote Keyless Entry systems
- Alarm systems
- Garage door openers
- Building door access
- Tire pressure monitors
- Meter reading
- Wireless sensor reading
- Low power radio telemetry

Device	Frequency	Modulation
rfPIC12F675K	290-350 MHz	ASK/FSK
rfPIC12F675F	380-450 MHz	ASK/FSK
rfPIC12F675H	850-930 MHz	ASK/FSK

Table 1 – rfPIC12F675 frequency ranges.

The rfRXD0420 is a UHF ASK/FSK receiver that is compatible with the rfPIC12F675F series of transmitters. The rfRXD0420 easily interfaces to any PIC micro enabling it to communicate with the rfPIC12F675F. Both the rfPIC12F675F and the rfRXD0420 are available only in a surface mount package. This can make prototyping a system a bit difficult. Fortunately, Microchip sells an inexpensive development kit called the rfPIC Development Kit 1 that includes the PICkit 1 Flash Starter Kit (Figure 1) as well as the rfPIC12F675F transmitter and rfRXD0420 receiver soldered onto modules for easier development. The PICkit 1 Flash Starter Kit includes an expansion header that accommodates both the transmitter and receiver modules for in-circuit programming and development.

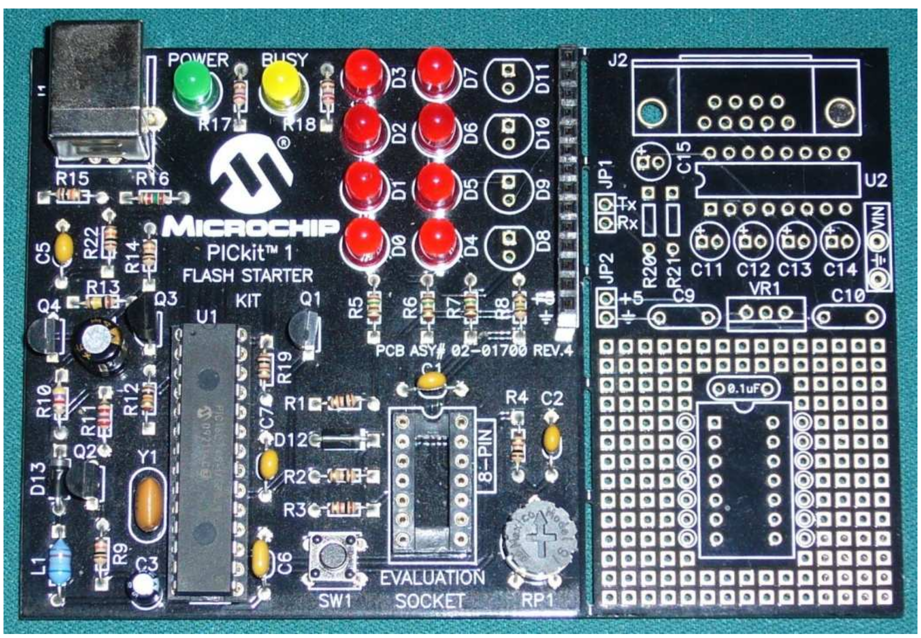

Figure 1 – PICkit 1 Flash Starter Kit

The transmitter module (Figure 2) includes the following:

- Two push-button switches connected to GP3 and GP4
- Two potentiometers connected to GP3 and GP4
- RF enable pin connected to GP5
- Data ASK pin connected to GP2
- Optional 8-pin socket for In-Circuit Emulation (ICE) or PIC12F675.

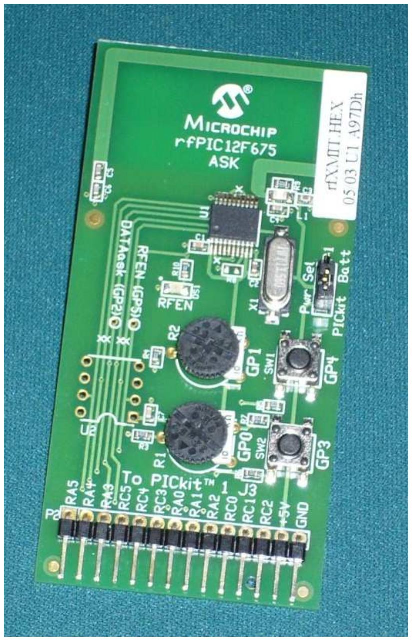

Figure 2 – rfPIC12F675 ASK Transmitter

The receiver module (Figure 3) includes the following:

- Single channel, fixed frequency at either 315MHz or 433.92MHz
- ASK Modulation
- 4800 baud signal rate

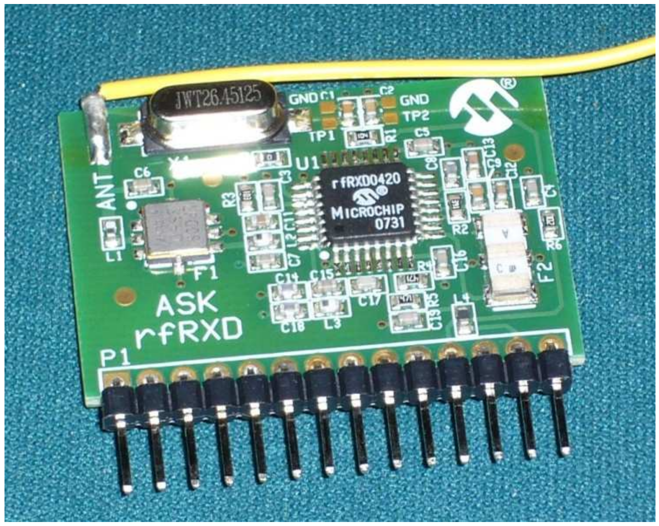

Figure 3 – rfRXD ASK receiver module

Because the rfPIC12F675F transmitter module is only available with ASK modulation the FSKout pin along with the DATAfsk pin are not used. All data output is modulated out of the DATAask pin. The antenna pin is tied to PCB traces that function the same as an external antenna. Switches GP3 and GP4 are tied to the RFENable pin along with an LED to indicate when the module is transmitting. Pressing G3 transmit the command to turn right, GP4 commands the robot to turn left and pressing both G3 and G4 commands the robot to go forward.

The synchronization protocol this projects uses to establish communication between the rfPIC12F675 and the receiver has been adapted from the getting started examples provided by Microchip. To establish communications, the rfPIC12F675 transmits a four part message with the data residing in the third part of the message. The first part is the *preamble* which consists of a series of sixteen .8ms Hi/Lo pulses followed by a *header* consisting of a 4ms Lo pulse. The purpose of the preamble and header is to notify the receiver that a message is on the way in-order to synchronize the start of the data bits. The third part of the message consists of the actual data to be transmitted. Each data bit is transmitted in the following manner:

HHHHHHHHLLLL for a 0 bit and HHHHLLLLLLLL for a 1 bit where H and L are .4ms long

Finally, the fourth part referred to as the *guard*, an 18.4ms Lo, completes the message.

The receiver module rfRXD0420 does not contain a microcontroller but is essentially an matching RF demodulator designed to interface to most any microcontroller. The rfRXD0420 consists of a minimal number of connections (Power, ground, antenna and data out). To control the robot, the rfRXD0420 is matched up with a PIC16F676 although most any PIC could be used since all the robot controller needs to do is receive the commands from the rfPIC transmitter and relay them to the servo controller. Figure 4 schematic shows the details of the robot receiver.

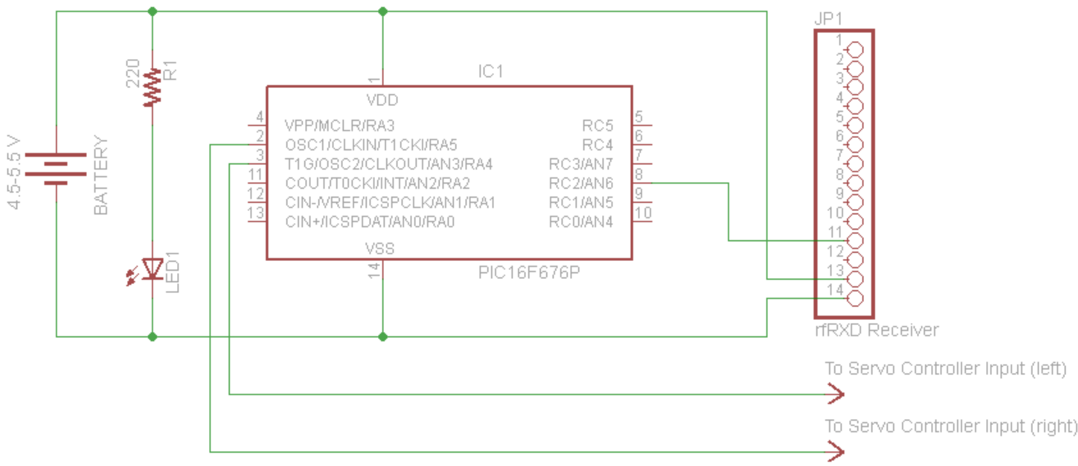

Figure 4 – Robot Receiver

III. Robot Platform Design

The robot chassis was constructed from an inexpensive kit from a company called Budget Robotics[2]. Budget Robotics offers a robot chassis called the ArdBot Chassis for just $14.95. The ArdBot Chassis is designed specifically for interfacing to microntrollers and the only additional hardware required is two wheels and servos. Other robot platforms can be used but the big bennefit of going with the ArdBot asside from price is the additional experience the students will receive from building the robot completely from the ground up. The completed robot with base, receiver and controller can be seen in Figure 5. The schematic diagram for the servo controller is shown in Figure 6.

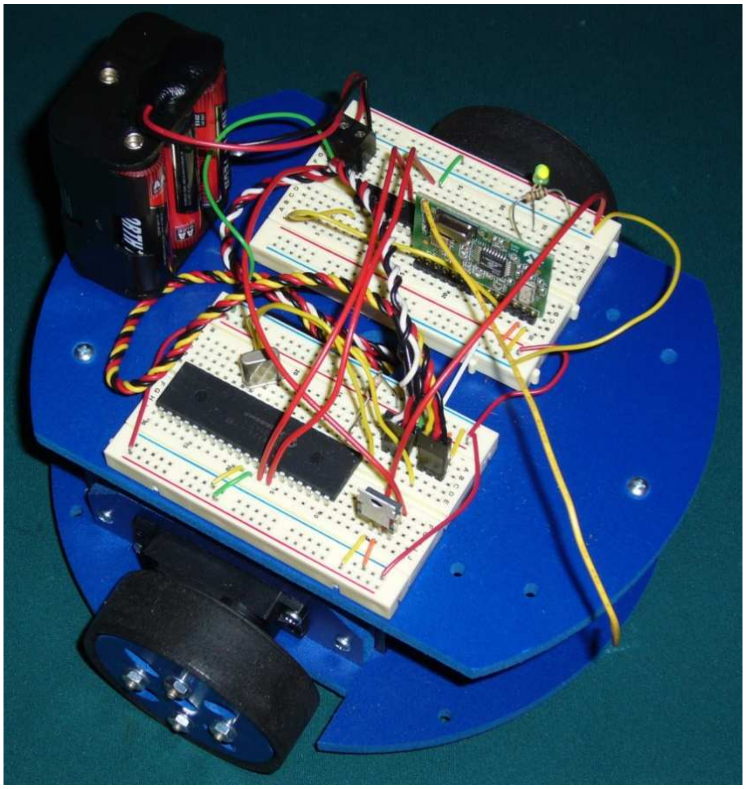

Figure 5 – Robot base, receiver and controller

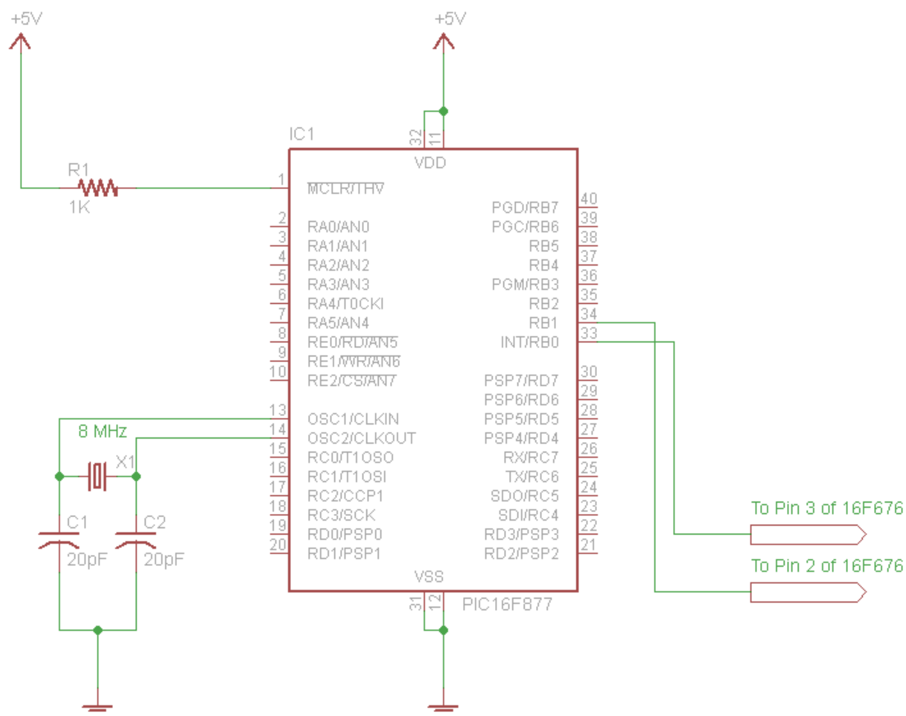

Figure 6. Schematic Diagram for the Servo Controller

The servos that drive the robot can be any standard servo but they must offer full rotation. These can be purcased from most places that servos or robot parts such as Parallax Inc[3]. A standard servo can also be modified to provide full rotation. Instructions on how to modify a servo for full rotation can be readily found on the Internet. As for the servo controller, we chose to go with a second microcontroller, a PIC16F877. The reason for choosing this microcontroller was simply because it was salvaged from another robot project. Another possiblity would be to use a BASIC Stamp for the servo control. All it needs to do is receive the direction command on two input ports and then enable the appropriate servo(s).

IV. Firmware Design

The transmitter firmware, written in combination of the free version Hi-Tech C and MPASM assembler, demonstrates some of the capabilities if the rfPIC12F675 by transmitting the status of the two buttons GP3 and GP4 on the transmitter[4 5]. A 0x23h is transmitted if the left button is pressed and a 0x43 is transmitted if the right button is pressed. The Xmit function used to transmit the data uses a simplified KeeLoq protocol based on the assembler code provided with the PICkit 1 and has been adapted to interface to the Hi-Tech C compiler. The Xmit functions are accomplished by the following code:

```
#include <htc.h>
#include <pic.h>
#include <picrf675f.h>

/*************************************************************
*  Data Memory Code Protection disabled (CPD_OFF)
*  Program Memory Code Protection disabled (CP_OFF)
*  Brown-out Detection disabled (BOREN_OFF)
*  MCLR pin function is I/O (MCLR_OFF)
*  Power-up Timer disabled (PWRTE_OFF)
*  Watchdog Timer disabled (WDTE_OFF)
*  Internal Oscillator (FOSC_INTRCCLK)
*************************************************************/
__CONFIG (INTIO & CPD_OFF & CP_OFF & BOREN_OFF & MCLRE_OFF & PWRTE_OFF & WDTE_OFF &
FOSC_INTRCCLK);

#define POT0        GPIO0       //not used
#define  POT1       GPIO1       //not used
#define  TXD        GPIO2       //serial data out
#define  BUTTON1 GPIO3          //switch 1
#define  BUTTON2 GPIO4          //switch 2
#define  RFENA      GPIO5       //transmitter enable

#ifndef _XTAL_FREQ
   #define _XTAL_FREQ 4000000          //4MHz system frequency
#endif

extern void xmit();

unsigned char w_temp;           //used for context saving
unsigned char status_temp;      //used for context saving
unsigned char TEMP;             //general purpose temporary register
unsigned char CSR0;             //TX buffer shift register
unsigned char CSR1;
unsigned char CSR2;
unsigned char CSR3;
unsigned char CSR4;
unsigned char CSR5;
unsigned char CSR6;
unsigned char CSR7;
unsigned char CSR8;

unsigned char Count;
unsigned char Count2;
unsigned char BitCount;
unsigned char TimeHi;
unsigned char TimeLo;
```

```
unsigned char FuncBits;            //function bits

void main()
{
    TRISIO = 0b00011011;
    ANSEL = 0;                     //disable A/D
    INTCON = 0b00001000;           //enable port change wakeup from sleep
    WPU = 0b00010000;              //pull-up on for GPIO4 (GPIO3 already set)
    IOC = 0b00011000;              //Wakup on change enabled for GPIO3 and GPIO4
    VRCON = 0;                     //disable internal Vref
    CMCON = 7;                     //disable comparators
    OPTION_REG = 0;                //Pull-ups on
    TXD = 0;                       //low for inverted serial out
    RFENA = 0;                     //RF transmitter off

    while(1) {

            if ((BUTTON1==1) && (BUTTON2==1)){
               RFENA = 0;          //disable transmitter
            }
            else {
               RFENA = 1;          //enable transmitter

          if(BUTTON1==1) {    //button 1 pressed
                  FuncBits = 0x23;
            xmit();
               }
               if(BUTTON2==1) {    //button 2 pressed
            FuncBits = 0x43;
            xmit();
          }
            }
    }
}

;  Name     : xmit.asm
;  Author   : Paul Lapsansky
;  Date     : 07/25/2011
;  Version  : 1.0

#include <aspic.h>

GLOBAL _xmit,_w_temp,_status_temp,_TEMP
GLOBAL _CSR0,_CSR1,_CSR2,_CSR3,_CSR4,_CSR5,_CSR6,_CSR7,_CSR8
GLOBAL _Count,_Count2,_BitCount,_TimeHi,_TimeLo,_FuncBits

psect text,local,class=CODE,delta=2
```

```
WaitxTE:
          movwf           _Count2

waitxlp:
          movlw           0x4f
          movwf           _Count

wait400lp:
          nop
          nop
          decfsz          _Count,F
          goto            wait400lp

          decfsz          _Count2,F
          goto            waitxlp

          retlw           0

_xmit:
          movlw           0x73                        ;send serial number
          BANKSEL         (_CSR0)
          movwf           BANKMASK(_CSR0)

          BANKSEL         (_FuncBits)
          movf            BANKMASK(_FuncBits),w       ;send function bits
          BANKSEL         (_CSR1)
          movwf           BANKMASK(_CSR1)

          bsf             0x03,0x05
          BANKSEL         (ADRESL)
          movf            BANKMASK(ADRESL),w
          bcf             0x03,0x05
          BANKSEL         (_CSR2)
          movwf           BANKMASK(_CSR2)

          BANKSEL (ADRESH)
          movf            BANKMASK(ADRESH),w
          BANKSEL         (_CSR3)
          movwf           BANKMASK(_CSR3)

          movlw           0x56                        ;send 32-bit serial number 1
          BANKSEL         (_CSR4)
          movwf           BANKMASK(_CSR4)

          movlw           0x34
          BANKSEL         (_CSR5)
          movwf           BANKMASK(_CSR5)
```

```
        movlw       0x12
        BANKSEL      (_CSR6)
        movwf       BANKMASK(_CSR6)

        movlw       0x20
        BANKSEL     (_CSR7)
        movwf       BANKMASK(_CSR7)

        movlw       0x55                ;send flags
        BANKSEL     (_CSR8)
        movwf       BANKMASK(_CSR8)

        movlw       0x10                ;init number of preamble bits
        BANKSEL     (_BitCount)
        movwf       BANKMASK(_BitCount)

Preamble:
        bsf         GPIO2               ;on
        movlw       1
        call        WaitxTE                     ;delay
        bcf         GPIO2               ;off
        movlw       1
        call        WaitxTE                     ;delay
        BANKSEL     (_BitCount)
        decfsz      BANKMASK(_BitCount),F       ;loop
        goto        Preamble

        movlw       0x0A                ;sync pause
        call        WaitxTE

        BANKSEL     (_CSR0)
        movlw       BANKMASK(_CSR0)     ;send 72 bit pattern
        movwf       FSR

TXNextByte:
        movlw       0x08
        BANKSEL     (_BitCount)
        movwf       BANKMASK(_BitCount)

TXNextBit:
        rrf         INDF,W              ;8 bit rotate
        rrf         INDF,F              ;carry contain LSB
        btfsc       3,0
        goto        ONE

        movlw  2                        ;
        BANKSEL     (_TimeHi)           ;
        movwf       BANKMASK(_TimeHi)   ;   +---+---+     +--
```

```
        movwf       1                       ;   |        |    |
        BANKSEL     (_TimeLo)               ;   |        |    |
        movwf       BANKMASK(_TimeLo)       ;---+        +---+
        goto        Trasm_BIT               ;   |  2Te   Te |

ONE:
        movlw       1                       ;
        BANKSEL     (_TimeHi)               ;
        movwf       BANKMASK(_TimeHi)       ;  +---+        +--
        movlw       2                       ;  |    |       |
        BANKSEL     (_TimeLo)               ;  |    |       |
        movwf       BANKMASK(_TimeLo)       ;---+  +-------+
                                            ;   | Te   2Te  |
Trasm_BIT:
        bsf         GPIO2                   ;on
        BANKSEL     (_TimeHi)
        movf        BANKMASK(_TimeHi),W
        call        WaitxTE

        bcf         GPIO2                   ;off
        BANKSEL     (_TimeLo)
        movf        BANKMASK(_TimeLo),W
        call        WaitxTE

        BANKSEL     (_BitCount)
        decfsz      BANKMASK(_BitCount),F
        goto        TXNextBit               ;loop on bits

        incf        FSR,F
        BANKSEL     (_CSR8)
        movlw       BANKMASK(_CSR8)+1       ;check if finished
        xorwf       FSR,W
        andlw       0x1F
        btfss       3,2
        goto        TXNextByte

        movlw       0x2E                    ;guard time
        call        WaitxTE

        return
```

The PIC16F676 receiver firmware is the unmodified receiver demo program that is included with the rfPIC Development Kit. This program outputs commands to the servo controller. The servo controller firmware is written in microEngineering Labs' PICBASIC Pro[6]. The servo control is accomplished with the following code:

```
Do
 If PORTB.0 = 1 And PORTB.1 = 1 THEN
   Low PORTC.0
   Low PORTC.1
   PulsOut PORTC.0,100
   Pulsout PORTC.1,200
   Pause 20
 Endif
 If PORTB.0 = 1 And PORTB.1 = 0 Then
   Low PORTC.0
   Low PORTC.1
   PulsOut PORTC.0,100
   Pulsout PORTC.1,100
   Pause 20
 Endif
 If PORTB.1 = 1 And PORTB.0 = 0 Then
   Low PORTC.0
   Low PORTC.1
   Pulsout PORTC.0,200
   Pulsout PORTC.1,200
   Pause 20
 Endif
Loop
```

The PICBASIC code compatible with the BASIC Stamp's P-BASIC making for the substitution of a BASIC Stamp for the servo controller quite simple.

V. Pedagogy of the Course

The pedagogy of the course is based on Outcome Based Education[7], and utilizes the interactive model of learning. All the students maintain an online portfolio of their work. The system designed in the laboratory to perform a specific task is the core measurement as the learning outcome of the course. The laboratory performance of the course is performed in teams of three students. This mode provides a platform for horizontal learning through active and engaged discourse and discussion. Students are empowered to charter their learning and feed their curiosity. Students are required to master the soft skills of comprehensive report writing on a weekly basis and of Technical Project Report writing and project oral presentation based upon the Team's Final Project. These classroom practices and laboratory environment provides a challenging and invigorating environment that prepares them for a lifelong learning process and career path.

VI. Outcome Measurement Rubric

The following outcomebased grading rubric was utilized in the measurement of learning objectives of the course and for course grade allocation.

Performed By	**Measured Item**	**Percentage Weight**	**Measurement**
Individual Student	15 Weekly Reports, covering the subject content mastered	30%	Content knowledge mastery and technical report writing.
Team of Three	15 Demostrations of Laboratory Works completed.	30%	Physical realisign and design of the content knowledge and team cooperation.
Team of Three	Final Project Demonstration, with formal oral presentation.	15%	Physical Design Realiztion, providing knowledge content to context and oral presentation.
Individual Student	Formal Final Project Report.	15%	Comprehensive technical writing skills.

Table 2. Outcome Measurement Rubric

VII. Conclusion

This paper has demonstrated how a course can be taught that provides the student both the experience with embedded RF system design as well as robot design and construction. This course provides PIC programming experience in both C, assember and BASIC. The programmings tools used here are available for download at no cost to the student and the required hardware including the rfPIC kit can be purchased for less than the cost of most other commercial robot kits. The flexibility of this course allows for other languages or compilers and microcontrollers to to be substituted if so desired. The design of this couses also leaves open a lot of area for improving the design of the robot including converting the receiver firmware to all Hi-Tech C code as well as combing the receiver and servo micrcontrollers with one single microcontroller.

Bibliography

[1] Microchip Technologies Data Sheets. http://wwwmicrochip.com

[2] Budget Robotics: PO Box 5821 Oceanside, CA 92502. http://www.budgetrobotics.com

[3] Parallax Inc.: 599 Menlo Drive Rocklin, CA 95765. http://www.parallax.com

[4] Sanchez, J., & Canton, M.P., *Microcontroller Programming the Microchip PIC*. CRC Press, 2007.

[5] Predko, Myke, *Programming and Customizing the PIC Microcontroller*. McGraw-Hill, 2008.

[6] MicoEngineering Labs, Inc.: 2845 Ore Mill Road, STE 4 Colorado Springs, CO 80904. http://melabs.com

[7] Omer Farook, Jai P. Agrawal, Chandra R. Sekhar, Essaid Bouktache, Ashfaq Ahmed and Mohammad Zahraee "Outcome Based Education And Assessment", Proceedings of the 2006 American Society for Engineering Education Annual Conference & Exposition June 20 -23, 2006. Chicago, IL.