INTRODUCTION

Early infrared (IR) remote control systems employed a simple on-off modulation of an IR LED to transmit the control data between the remote control and the appliance. Unfortunately, while this method was simple to implement, it suffered from both interference and range problems. A warm object, or a reflection of the sun in a window could, and often did, mask the IR energy received from the transmitter, reducing its effective range. In addition, mechanical modulation of an IR source (for example, air moving a window shade) often resulted in the reception of garbled or erroneous commands by the receiver.

To correct these problems, modern IR control systems are based on a modulated 38 kHz carrier. Command data is transmitted by on-off keying the 38 kHz modulation of the IR LED, resulting in variable length packets of 38 kHz pulses (see Figure 1). Using a modulated carrier frequency gives IR remote controls several advantages:

1. A modulated carrier frequency can be amplified by AC coupled amplifiers in the receiving circuitry. The AC coupling passes the modulated signal, but not DC shifts due to continuous IR sources.
2. A modulation frequency of 38 kHz is faster than the movement of most, if not all, mechanical systems, so unintentional modulation of a passive IR source is less likely to cause interference.
3. Finally, 38 kHz is low enough in frequency to allow low cost bandpass filtering of the carrier frequencies. Bandpass filtering of the carrier allows the use of multiple modulation frequencies which, in turn, makes possible the simultaneous use of multiple remotes without crosstalk between systems.

The system described in this Technical Brief implements a modulated 38 kHz IR driver with minimal firmware overhead and good overall efficiency. The driver is compatible with most off-the-shelf, inexpensive, 38 kHz receiver modules.

THEORY OF OPERATION

Typical low cost IR remote controls use a modulated linear current source to bias the IR LED. Whenever the LED is driven, the battery voltage is reduced to the IR LED’s forward voltage at the desired bias current by a current limiting resistor. Unfortunately, this means that some of the battery power is lost as heat in the resistor. Specifically, the difference between the battery voltage and the forward voltage of the LED multiplied by the bias current. Given the forward voltage of the diode is typically 2 volts and the battery is 3 volts, this means that as much as 1/3 of the energy from the battery is wasted as heat whenever the LED is driven.

A more efficient method of energy transfer is to use a switching power supply technique (see Figure 2). During the first half of the pulse cycle, energy is built up in the inductor as a magnetic field. During the second half of the cycle, the inductor is switched across the LED and the energy is discharged into the LED. Because the inductor cannot change its current flow instantaneously, the peak current flow created during the first half of the cycle is the same as the peak current flow delivered during the second half. In addition, the voltage developed by the inductor during the second half of the cycle will be exactly the forward voltage of the series diodes, eliminating the loss of a limiting resistor.

Equation 1 gives the relationship between supply voltage, charge time, current and inductance.

\[ I = \frac{V_{SDT}}{L} \]

**WHERE**

- \( I \) = CURRENT
- \( V_S \) = SUPPLY VOLTAGE
- \( D \) = CHARGE DUTY CYCLE
- \( T \) = PERIOD
- \( L \) = INDUCTANCE

Due to conservation of energy, some compensation must occur to balance the energy in and out of the system. If the charge cycle builds up a 100 mA flow from a +3 volt DC supply and then discharges 100 mA into a +2 volt DC load, then energy is lost, right? Actually, the time to charge and time to discharge charge to compensate for the different charge and discharge voltages. If the charge cycle takes 2 \( \mu \)S at +3 volts, then the discharge will take 3 \( \mu \)S at +2 volts (see Equation 2).
CIRCUIT DESCRIPTION

In the example circuit, the Programmable Switch Mode Controller (PSMC) in the PIC16C782 will be used as the controlling peripheral. The PSMC is uniquely suited to this function as the pulse start timing is locked to the MCU system clock, and the pulse width is controlled by analog feedback. This means the accuracy of the pulse timing is very precise, while the pulse width can be tailored via analog feedback to provide the required current flow regardless of the system supply voltage (see Figure 2).

The circuit is broken into four main sections:
1. Frequency control.
2. Energy transfer.
4. Data pulse width control.

FREQUENCY CONTROL

The frequency of the output pulses from the PSMC are selected by the MCU clock frequency and the prescaler setting of the PSMC. Given the 4 PSMC prescaler options, a 38 kHz pulse output requires that the MCU clock frequency be a 16, 32, 64, or 128 multiple of 38 kHz. In the design example, a clock frequency of 2.432 MHz was chosen (see Table 1).
To simplify obtaining a crystal, a 2.4576 MHz crystal was used. The IR receivers typically have a percentage bandwidth of ±5%, so the difference of 1% between a 2.4576 MHz crystal and the calculated value of 2.43 MHz (64*38 kHz) should not adversely affect the transmission between the driver and the receiver. However, if an RC oscillator is chosen as the clock source for the MCU, its frequency drift with temperature and the receiver’s bandwidth should be considered when estimating the reliability of data transmission.

ENERGY TRANSFER

At the beginning of each cycle, the PSMC starts a pulse, (presuming that the comparator output is high) by turning on Q1. This effectively shorts L1 across the supply voltage. Initially, the current through L1 will be zero, but as the pulse time continues, the current through L1 will increase linearly (see Equation 1). Proportional to the current in L1, the voltage across RSENSE will also increase proportionally. When the voltage across RSENSE equals the reference voltage present at the non-inverting input of the comparator, the comparator output goes low and the PSMC ends the charge pulse. With Q1 now off, the current flow in L1 must find another conduction path. In an effort to continue the flow of current, the voltage across L1 increases until diodes D1 through D4 forward bias and conduct. Once the diodes conduct, the energy in L1 is discharged into the light emitting diodes. When the energy has been discharged the system is idle until the start of the next pulse.

A side benefit of using an inductive energy transfer from the battery to the LEDs is that variations in the battery voltage have little effect on the energy transferred, because the charge cycle is held until the inductor current reaches the specified value, the energy transferred is held constant. If the battery voltage is low, the charge time of the inductor will lengthen to compensate for the lower charging voltage. A minor phase shift is incurred due to the longer charge time, but this does not affect the frequency of the transfer and has no effect on the integrity of the transmission.

ON-OFF MODULATION OF THE CARRIER

On-Off modulation of the 38 kHz carrier is accomplished by simply enabling and disabling the PSMC module. Safeguards within the PSMC automatically complete any pulse in progress and then disable the module when the pulse is complete.

Once disabled, control of the I/O pin will revert to the next highest priority peripheral. If the voltage comparator C1 has its output enabled, then C1 will take control of the pin. If the PSMC and C1 are not enabled, then control reverts to the port driver RB6. To prevent damage to the MOSFET, two precautions must be taken:

- The port driver for pin RB6 must be configured as a digital output driving a logic 0 output.
- The output for C1 must be disabled. This will hold the MOSFET in its off state when the PSMC is disabled.

Care must also be taken to ensure that no Read-Modify-Write operations are performed on PORTB, as this can inadvertently set RB6 if the instruction is performed while the PSMC is driving the pin high as part of an output pulse.

### TABLE 1: FREQUENCY OF OUTPUT PULSES FROM PSMC

<table>
<thead>
<tr>
<th>Pulse Frequency</th>
<th>PSMC Prescaler</th>
<th>CPU FOSC (calculated)</th>
</tr>
</thead>
<tbody>
<tr>
<td>38 kHz</td>
<td>16</td>
<td>608 kHz</td>
</tr>
<tr>
<td>38 kHz</td>
<td>32</td>
<td>1.216 MHz</td>
</tr>
<tr>
<td>38 kHz</td>
<td>64</td>
<td>2.432 MHz</td>
</tr>
<tr>
<td>38 kHz</td>
<td>128</td>
<td>4.864 MHz</td>
</tr>
</tbody>
</table>
CONTROL OF DATA PULSE WIDTH:

Most IR remote control systems transmit control data by keying the 38 kHz carrier with one of several specific length pulse bursts. Therefore, it is important to be able to accurately control the number of pulses in each packet generated by the transmitter.

The example design accomplishes control of its packet width by connecting the PSMC pulse output to the Timer0 clock input. Each time a pulse is generated by the PSMC, Timer0 is incremented. To generate a specific number of pulses, the firmware has only to load Timer0 with a value equal to 255 minus the number of pulses required, enable the PSMC, and then monitor the Timer0 Interrupt Flag for the rollover of Timer0 from FF to 00. As soon as the rollover occurs, the PSMC is disabled and the pulse packet is terminated. In firmware, the rollover condition could either be polled, or enabled to generate an interrupt.
DEMONSTRATION FIRMWARE

The attached firmware implements the IR serial protocol presented in Figure 1. It is composed of two files, **IR_DRV.R.INC** which is a setup and configuration file generated by the PICDEM MSC1 Graphical User Interface (GUI) program, and **IR_TX.ASM** which contains the control and serializing routines for the firmware.

The setup and configuration firmware in **IR_DRV.R.INC** performs the following:

1. PORTA:
   - RA4 as the Timer0) Clock input
   - RA3 as the VREF1 input to C1.

2. PORTB:
   - RB0 as the VR module output
   - RB2 as the current sense feedback input
   - RB6 as the PSMC1A output.
   - RB6 [Data] = 0, to hold the MOSFET off when the PSMC is disabled.

3. VR module:
   - Module enabled.
   - Output enabled. (see PORTB)

4. Comparator C1:
   - Enabled
   - No output
   - Output non-inverted
   - VREF1 on the non-inverting input
   - RB2 for the inverting input
   - Normal speed.

5. Timer0:
   - T0CKI clock input
   - Positive edge
   - 1:1 prescaler

6. PSMC:
   - 64:1 Prescaler
   - Single output
   - Non-inverted output
   - PWM mode
   - Minimum duty cycle 0%
   - Maximum duty cycle 50%
   - C2 input disabled

6. Configuration Word:
   - EC oscillator, external clock input
   - External MCLR
   - No code protect
   - No WDT

The main firmware is contained in **IR_TX.ASM**. The file contains routines to format and transmit a multi-byte ASCII message through the IR transmitter.

The **MAIN** section of the program performs two functions: 1. It calls the **HEADER** routine to preface each transmission with 16 header pulses, and 2. It sequentially transmits the 11-character string stored in its table.

The **HEADER** routine generates a string of sixteen groups of 200 pulses, separately by dark spaces, which act as a header and synchronization signal for the receiver. Between the groups of 200, a standard 50 cycle dark period is also generated to separate the header pulses.

The **TRANSMIT** routine accepts an 8-bit value from the W register and serially transmits the data using the pulse format described in Figure 1.

**TX_ON** and **TX_OFF** routines handle enabling and disabling the PSMC for the ON-Off modulation of the 38 kHz signal.

Finally, **DELAY** and **OUTPUT** perform the generation of the 38 kHz pulse groups and dark periods of each transmission. **DELAY** configures Timer0 to operate from the internal clock FOSC/4 and generates a dark period equal to a 50 cycle group of 38 kHz pulses. **OUTPUT** configures Timer0 to operate from its external clock input T0CKI and generates 38 kHz pulse groups based on the width value passed to the routine through the W register.

Together, the two files transmit a sixteen-bit header followed by 11 ASCII characters to form the message: “HI PWR IRTX.”

BENEFITS

The major benefits of this design are:

1. Efficient transfer of energy from the battery supply to the IR LEDs.
2. Automated generation of the 38 kHz carrier frequency.
3. Simple control of the pulse packet size.
4. Minimum parts count.
5. Constant LED power, even during battery voltage roll-off at end-of-life.
CONCLUSION

Prior implementations of a 38 kHz IR remote control system have often suffered from inefficient energy transfer from the battery to the IR LEDs and required software-intensive control. The example presented here improves the energy transfer efficiency and significantly simplifies the generation and modulation of the control data onto the 38 kHz carrier.

The protocol used in this example was not modeled after any manufacturers protocol, however, most protocols are based on a variable packet length format. To modify this protocol for a specific manufacturer is reasonably simple. Just measure the packets used by the manufacturer and modify the delay, transmit and output routines to send the new packet sizes.

<table>
<thead>
<tr>
<th>TABLE 2: MEMORY USAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program Space</td>
</tr>
<tr>
<td>RAM</td>
</tr>
</tbody>
</table>
Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable".
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.
WORLDWIDE SALES AND SERVICE