SOFTWARE STATE MACHINE DECODER FOR THE INFRARED REMOTE CONTROL UNITS USING BI-PHASE CODING SCHEME

Boris Ribov, Ph.D. Student, Eng., LZ1BJR

Space Research Institute - Bulgarian Academy of Sciences, 6 Moskovska Street P.O. Box 799, Sofia 1000, Bulgaria, phone: +359 88 7 261 866, e-mail: ribov@developer.bg

Valentin Vasilev, M.Sc., Eng.

Department of Research & Development, VALVECS HOLDING PLC, 150 Hristo Botev blvd, Plovdiv 4000, Bulgaria, phone: +359 32 620302, e-mail: vv@valvecs.com

Keywords: IR, Remote control unit, Bi-Phase code, Pulse coded, Space coded

Today, the remote control for electronic devices has become very ordinary. The widespread use of remote control units for TVs, VCRs, and HiFi has been followed by its use for air conditioners, opening doors, and portable player control, etc. Contemporary needs for the stability of the infrared transmission protocol are increased. At the same time the efforts are direct to lower the production price and integration price. Therefore most of the producers prefer to implement software IR decoder in the main microprocessor for the remote control receiver, instead of hardware integrated circuits.

Here we present a robust software infrared decoder based on the state machine to decode the bi-phase channel code. Bi-Phase code has better performance than other code schemes. But there is no easy way to extract the data from the channel code compared to the other coding schemes like "pulse coded signal" and "space coded signal".

This paper describes implementation of the software state machine IR receiver into the ST20 Embeded Real Time Operation System (ST20 RTOS) for use in contemporary applications like CD/VCD/DVD players, DVB SAT receivers, etc. Our software decoder was developed using high-level language "C", so it is possible to integrate into different hardware platforms, which is compatible with C code and RTOS principle.

1. INTRODUCTION

There are two different standards for remote control communication. These standards are REC-80 [4] and RC-5 [4]. The REC-80 standard varies the length of the pulses that are sent out. Different pulses signify different binary values.

Figure 1 – REC-80 coding scheme

There are two versions of the REC-80 standard. The first version, pulse coding, varies the length of the pulse of light. One pulse is interpreted to be a '0' while two pulses are interpreted to be a '1'. The duration of the space between two pulses is constant and is equal to the duration of one pulse. The other version, space coding, varies the length of the spaces between pulses. This works that same as the above except that everything is inverted. A '0' is defined as one space and a '1' is defined as two spaces. Here a pulse is used between two consecutive bits. The diagrams below show examples of both pulse and space coded signals.

The second standard is RC-5. This communication protocol is based on signal transitions during a prescribed amount of time. When the signal changes from high to low during this time is interpreted as a '0', when it goes from low to high it is said to be a '1'. This coding scheme is almost known as bi-phase coding or Manchester coding. The first diagram shows what the values of the two different transitions are while the second is a short example.

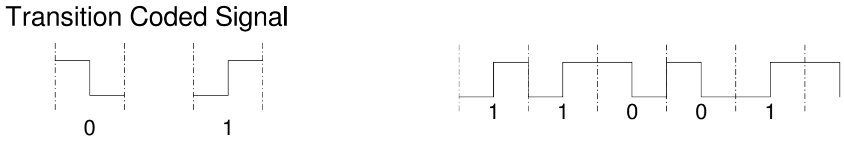


Figure 2 – RC-5 coding scheme

When a stream of data is sent from a remote control it is sent in a stand format. First there is a header, then an address, and then a command. The header is just a long pulse that turns on the receiver and is sent out before every code. The address field control which type of device the remote controls, e.g. TV, VCR, CD player. The command field is the actual command for the device. These commands vary from device to device and there is really no defined set of commands.

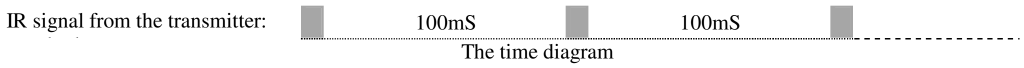
The decoding algorithm for the pulse coded signal and space coded signal is quite easy. Because of the constant marker at the beginning of each bit /pulse or space depending on the scheme/ it is easy to measure time interval between two markers and to decide the value encoded with this interval. So if we have "space coded" signal and the interval between the two high-level mark bits is short then we emit "0", if this interval is long then we emit "1". If the time exceeds the maximum time for a symbol transmission then we have a timeout or error in the protocol. If the time is smallest than symbol time then we have an error in the transmission protocol. If we have a timeout this means that the last bit was transmitted.

The errors appear due to the interference in the media or presence of multipath signals. Pulse coded signals and space coded signals are unstable to the multipath signals. One of the reasons is that the AGC (Automatic Gain Control) in the IR detector has wide time-constant referred to the bi-phase method. In the indoor infrared communications the presence of multipath signals is common due to the reflection from the walls and other objects in the room. There exist some coding schemes reducing the ISI (Inter-Symbol-Interference). One of them is known as CCK (Complimentary Code Keying). CCK [2] has better performance in the multipath environments. The other way to increase the stability is to use redundancy codes or some kinds of spread spectrum modulations like DSSS (Direct Sequence Spread Spectrum). DSSS [1] is a method based on 11-bits redundancy code named Barker's code. This code has best correlation properties and has good performance when we intent to transmit data below the noise level of the channel.

CCK and DSSS are complex code schemes. The integration price for these two techniques will be much more than bi-phase code scheme. That's why we decide to use bi-phase code for our experiments.

2. RCU PROTOCOL

The type of transmission is UNI-directional continuous IR transmission. When we hold the button on the remote control the frames are transmit every 100mS. The IR frame structure has several fields shown in figure 3.



1	**C5**	**C4**	**C3**	**C2**	**C1**	**C0**	**B0**	**D2**	**D1**	**D0**	**A2**	**A1**	**A0**	**R0**	**F7**	**F6**	**F5**	**F4**	**F3**	**F2**	**F1**	**F0**
STA	Custom ID						BAT	Device			Address			REP	Function							

Figure 3 – Frame structure

STA is the start bit, which is always 1. **C5-C0** is a custom ID number representing the Model_No/Device_No of the RC. **BAT** field is used to indicate battery low condition. It is 1 when the battery voltage is OK and is 0 when the battery voltage is low or critical. **D2-D0** represents device ID, so one RCU (Remote Control Unit) may be used to control up to 8 devices (for example TVs, VCRs, DVDs etc.). **A2-A0** is 3-bit hardware address of the chip in the remote control. **REP** represents a repeat condition. It is 0 in the first frame and is 1 in the repeated frames (when we hold the RCU button). **F7-F0** corresponds to the pressed button code. This is 8-bit code so we may control up to 256 buttons. The size of the whole frame is 23 bits and the repeat time of the frame is 100mS.

3. STATE MACHINE DIAGRAM

The state machine takes categorized pulse/space events as input and generates 23-bit control codes as output. When the state machine is started it emits a 1, and enters the **M1** state. Emitted bits are left-shifted into a 23-bit data store. Any event not illustrated as a state transition results in an error, at which point the decoder should reset and restart.

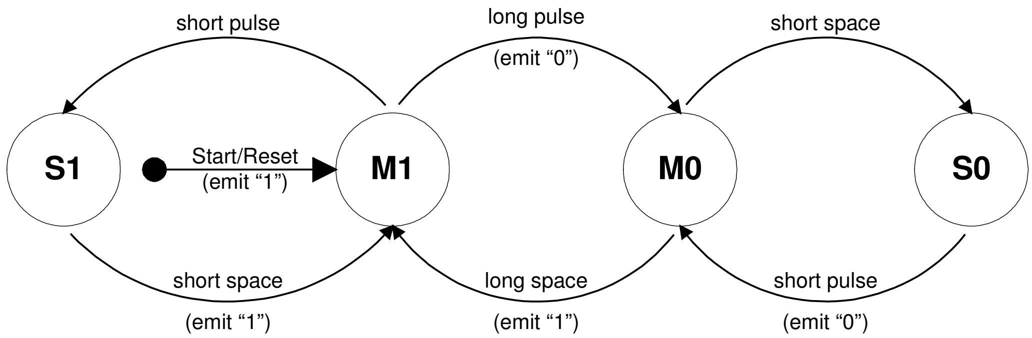


Figure 4 – State Machine Diagram

Decoding is complete when 23 bits have been emitted. It is also useful to continue decoding until the decoder is in state **S1** or state **M0**, to ensure the final pulse is consumed.

The state names reflect the position within the bi-phase signal. For example "M1" represents the middle of a '1' bit, after the pulse and before the space. The signal always change states at the middle of a bit, and the state machine always emits a decoded bit as it enters the mid-bit state.

The pulse/space events can be generate by the interrupt system and the timers inside the CPU. Then it is not necessary to monitor the IR input always and this release the CPU from this task. When an interrupt occurs on signal change on the IR port we can calculate the event type – short pulse, long pulse, short space, long space error or timeout by the current values stored in the registers of the hardware timer. When we categorize the event we need to send it to the state machine.

4. IMPLEMENTATION

We used RTOS – OS20 for the ST20 based chipsets and the high level ST-API drivers to capture data from the IR port. Infrared driver handles the port with the associated interrupt and returns parameters of the captured data – time duration of the captured pulses. When we use '*pulse coded signal*' and '*space coded signal*' this driver is enough for the IR decoding process but when we use bi-phase coding scheme this driver works like event capture for the state machine decoder. Here is the basic algorithm:

```
while (1)
{
   < Starting process to Capture IR data in the SymbolBuffer with specific timeout for the IR driver >

   semaphore_wait(&TestSemaphoreRead);

   reset_state_machine ();
   Code_Ready = false;

   for (i = 0; i < (SymbolsRead - 1); i++) {
               if (decode_state_machine (PULSE, SymbolBuffer[i].MarkPeriod ) == -1) {
                        reset_state_machine ();
                        continue;
               }
               if (s_m.curr_bit == 23) {Code_Ready = true; break;}

               if (decode_state_machine (PAUSE, SymbolBuffer[i].SpacePeriod) == -1) {
                        reset_state_machine ();
                        continue;
               }
               if (s_m.curr_bit == 23) {Code_Ready = true; break;}
   }
   if (Code_Ready == true) {
       iNewRemoteKeyVal = s_m.key_code;
       STTBX_Print(("The whole Code is %x\n",iNewRemoteKeyVal));
   }
}    /* end of while */
```

Because of the Real Time Operational System we have infinite loops. At the top of every iteration we initialize the capture driver with some parameters: *buffer* *pointer* – contain captured data of the IR pulses, *timeout* – specifies the maximum time of 'transition coded signal', *semaphore* – specifies the handle for the capture process, *max_buffer* – specifies the maximal number of symbols to record. When we start the capture driver then the processor is focus to the other processes in the OS kernel until we have an event from the IR capture driver to the *semaphore_wait* function, which is a part of the RTOS. This shows that we have critical event to process from the IR driver. This may be one of the following: *timeout occurs and there is captured data* or *timeout occurs and there is no captured data*. The first situation is typical when we press a key on the Remote Control Unit and the timeout occurs after the last transmitted bit from the RCU. The variable *SymbolsRead* contains the number of actual captured IR events/bits. If the value of this variable is zero it means that the timeout occurs but there was no IR activity. So in this situation we re-initialize the process by going to the next iteration in the infinite loop.

In the body of the algorithm we call the state machine decoder function (*decode_state_machine*) by giving the event parameters – pulse or pause event and the duration of the event. The decoder function checks the last state of the state machine and returns an error if there is a state that was not described. Other function named *reset_state_machine* sets the state machine in the initial state. To control the state of the machine we use global variables: *s_m.curr_bit* – decoded bit counter, *s_m.key_code* – contains the current meaning of the restored codeword. The variable *iNewRemoteKeyVal* contains the final code from the remote control.

5. PERFORMANCE

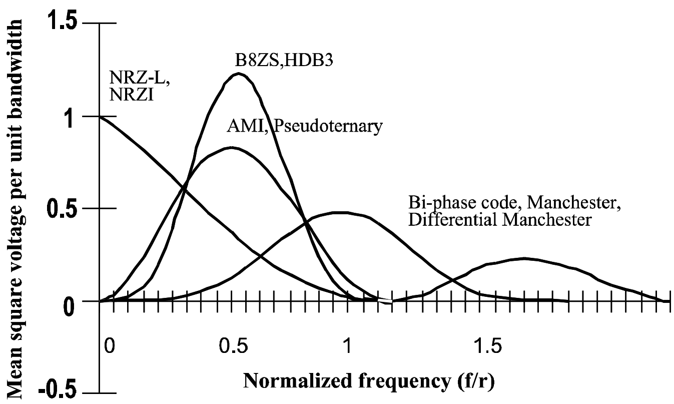


Figure 5 – Spectral density

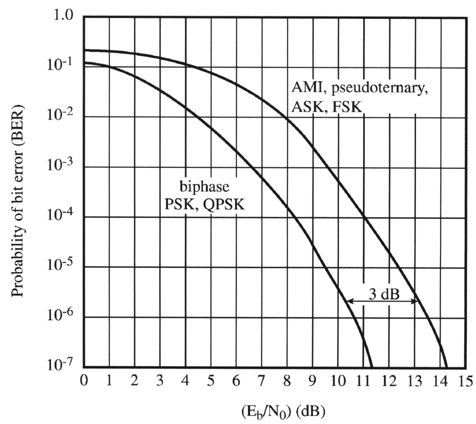


Figure 6 – BER performance

The power of the spectral density of the Bi-phase and Manchester code is not high (Figure 5). Thus increase the performance of the code. Some properties of the bi-phase code are: allows for clocking mechanism for both kinds of bits, the spectral density graph shows that there is not a dc component and the bandwidth is relatively narrow, noise on the line has to invert the signal before and after the inverted bit to avoid detection. The Bit-Error Rate (BER) performance (Figure 6) of the bi-phase code is better than ASK and FSK. There are another methods for transmitting in wireless diffuse optical channels named *Position Pulse Modulation* (PPM) and *Time*

Pulse Modulation (TPM). The comparison between them and the *On-Off-Keying* (OOK) [3] shows that PPM has better efficiency in diffuse infrared channels (Dir).

6. CONCLUSION

Primary this investigation was relevant to the plan of the Space Research Institute at Bulgarian Academy of Sciences – the topic is: "*Methods and resources for planning the aero-space communication information and navigation systems*". The specific problem studied in this publication was the ability for the integration of the software decoder for the short-range wireless communication channels. Described information relates to the Infrared communication channels but it can be used for the wireless radio-channels, too.

The results confirm the ability of the integration of low-cost robust method for decoding the data using the bi-phase coding scheme in the real-time operational systems. The bi-phase coding scheme produce better stability than the timed coded signal (nevertheless pulse or space coding). The CPU usage time of the algorithm is too small because the algorithm is based on interrupts and events. So this does not decrease the compute performance of the system at all.

The results shows that the bi-phase code is suitable for the wireless radio communication, because there is no DC component in the spectrum and it has good clocking mechanism. This application is very significant because infrared remote controls work in the area of direct visibility, instead of this radio waves can pass through obstacles.

7. REFERENCES

[1] Spasov G., Ribov B., "*Spread Spectrum for Wireless Systems using PIC based micro-controllers as a Spreader*", Proceedings of the Eleventh International Scientific and Applied Science Conference, ELECTRONICS ET'2002, Sozopol, September 25-27, Bulgaria, book 2, ISBN:954-438-327-1, pp.89-94, 2002

[2] Ribov B., Spasov G., "*Complementary Code Keying with PIC based Microcontrollers for Wireless Radio Communications*", International Conference on Computer Systems and Technologies, CompSysTech'2003, Sophia, June 19-20, Bulgaria, ACM Press - New York, NY, USA, ISBN:954-9641-33-3, pp.66-70, 2003

[3] Spasov G., Petrova G., *"Power and bandwidth Efficiency in Wireless Diffuse Optical Transmission Channel with Time Pulse Modulation"*, Proceedings of the Tenth International Scientific and Applied Science Conference, ELECTRONICS ET'2001, Sozopol, September 26-28, Bulgaria, book 3, ISBN:954-438-260-7, pp.27-32, 2001

[4] Ulmer B., McLean J., *Infrared in Embedded Systems*, April 15, 1998, http://www.egr.msu.edu/classes/ece482/Teams/98spr/design3/ap_stuff/my cats name is mittens/apnote/tr1-rpt.doc