Interfacing the C328 UART JPEG Camera to the AT91EB55

Zachary Byrne

April 12, 2010

The Interface

The camera module has a serial connection and uses an RS-232 style of communication at a voltage of 3.3V. The camera's command set and functionality are described very well in the module user manual, so this document will instead focus on the AT91EB55 side of the system as well as some gotchas that we encountered.

Command Structure

Each command is a series of 6 bytes which consists of a header (always 0xAA), a command id, and 4 parameters. The commands and their parameters are discussed in the user manual.

Sending Commands

The AT91 library contains functions specifically for sending multiple bytes over the usart. for sending commands, the at91_usart_send_frame() function will transmit full buffers over the usart instead of single bytes.

Receiving Commands

Like at91_usart_send_frame(), there is also at91_usart_receive_frame(). This function stores incoming data in a buffer directly. In this case oversived buffers are recommended as the data can sometimes become offset in the buffer.

Synchronizing the Camera

Before any pictures can be taken, the camera's serial connection must be synchronized with the evaluation board's usart. As described in the user manual sychronization is completed by repeatedly sending the SYNC command to the camera and waiting for it to acknowledge. This can be accomplished with the following code.

```
#define CMD_SIZE 6
#define SYNC_ATTEMPTS 60
#define BAUDS115200 ( 32000000 / ( 16 * 115200 ) )

u_int i, status;
char * buffer;
char sync[CMD_SIZE] = { 0xAA, 0x0D, 0x00, 0x00, 0x00, 0x00 };
char ack[CMD_SIZE]  = { 0xAA, 0x0E, 0x0D, 0x00, 0x00, 0x00 };

/*Create the buffer*/
buffer = malloc ( 2 * CMD_SIZE ); /*oversized buffer in case of offset*/
memset ( buffer, 0, 2 * CMD_SIZE );

/*Open the usart*/
at91_usart_open ( & USART2_DESC, US_ASYNC_MODE, BAUDS115200, 0 );
```

```
/*Set up the receive buffer*/
at91_usart_receive_frame ( & USART2_DESC, buffer, 2 * CMD_SIZE, 20 );

/*Send the sync command*/
/*Camera needs sync sent up to 60 times to detect baudrate*/
for ( i = 0 ; i < SYNC_ATTEMPTS ; i++ )
  {
    /*Send SYNC*/
    at91_usart_send_frame ( & USART2_DESC, sync, CMD_SIZE );
    printf ( ''sync attempt %d\n'', i );

    status = at91_usart_get_status ( & USART2_DESC );

    /*Check if something was received*/
    if ( status & ( US_ENDRX | US_TIMEOUT ) )
      {

        /*Try to get the frames*/
        if ( buffer[1] == 0x0E
             && buffer[2] == 0x0D
             && buffer[7] == 0x0D )
          {
            /*Send ACK*/
            at91_usart_send_frame ( & USART2_DESC, ack, CMD_SIZE );
            break;
          }
      }
  }
free ( buffer );
```

Initializing the Camera and Other Similar Commands

Many of the commands that can be sent to the camera module simply respond with an acknowledgement packet. These commands can be controlled in much the same way as synchronizing. The difference in this case is that the command is sent once and the status register check is placed in an infinite while loop. This could be accomplished like this:

```
#define CMD_ATTEMPTS 5

/*Attempt to send the command several times*/
for ( i = 0 ; i < CMD_ATTEMPTS ; i++ )
{
  /*Set up the receive buffer*/
  at91_usart_receive_frame ( & USART2_DESC, buffer, 2 * CMD_SIZE, 20 );

  /*Send command*/
  at91_usart_send_frame ( & USART2_DESC, command, CMD_SIZE );

  while ( 1 )
  {
    status = at91_usart_get_status ( & USART2_DESC );

    /*Check if something was received*/
    if ( status & ( US_ENDRX | US_TIMEOUT ) )
    {

      /*Try to get the frame*/
      if ( buffer[1] == 0x0E && buffer[2] == CMD_ID )
      {
        return TRUE;
      }
      break;
    }
  }
}
```

One thing to note is that although the user manual only lists preview resolutions up to 160x120, using the same values as the jpeg resolutions for 320x240 and 640x480 are valid parameters for preview images as well.

Capturing Pictures

The C328 has several different capture modes outlined in the user manual. The two that affect the way you interact with the module are whether the image is compressed or uncompressed. Uncompressed images are received from the camera in a single transmission unlike compressed images which are received in packets. The user manual shows examples of both modes and the code examples above are a decent starting point. One thing to mention is that the default heap size in the linking scripts provided in the lab is only 30 KB, where a 160x120 image in

16 bit RGB is roughly 40 KB, so the heap will need to be bigger or you can add some extra ram.

Gotchas

While reading up on these modules online I saw a few reoccuring issues people were having. Most of them were a result of not reading the maunal, but I did see one or two of them come up.

Synching

Sometimes the cameras simply will not sync. It happens about one time in twenty, but the only thing you can do is cycle the power and try again.

High Resolution JPEGS

Whenever I would try to initialize the camera with a JPEG resolution of 640x480 the camera would start returning NAKs indicating a parameter error. I did not search for a fix since we wanted to work with uncompressed images anyways.