RSV5-C328 Serial Camera Programming Instructions and Sample Code for Arduino

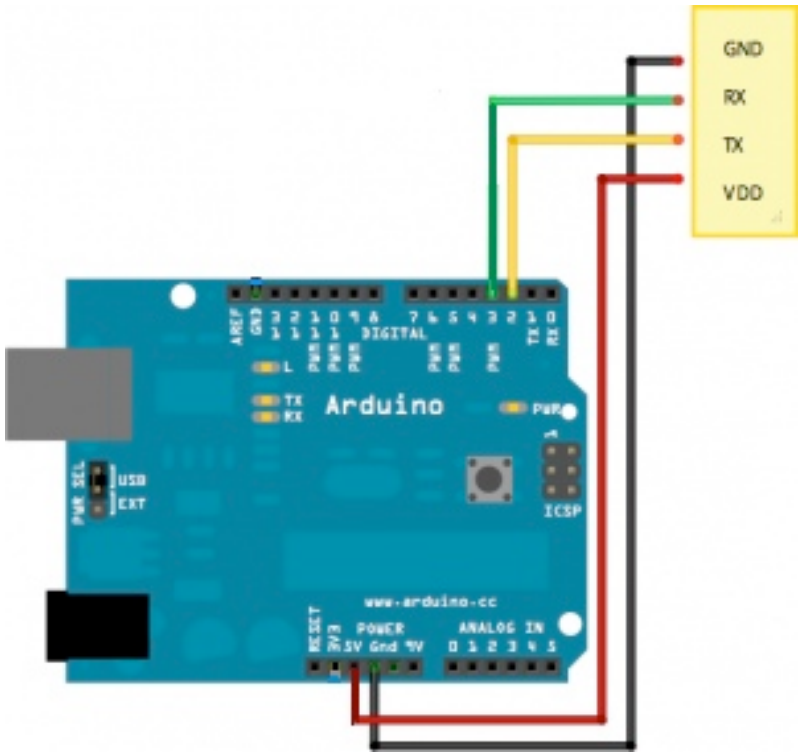
The camera module has an on-board serial interface (TTL or RS232) that is suitable for a direct connection to any host micro-controller UART or a PC system COM port. The camera is powered by 5v.

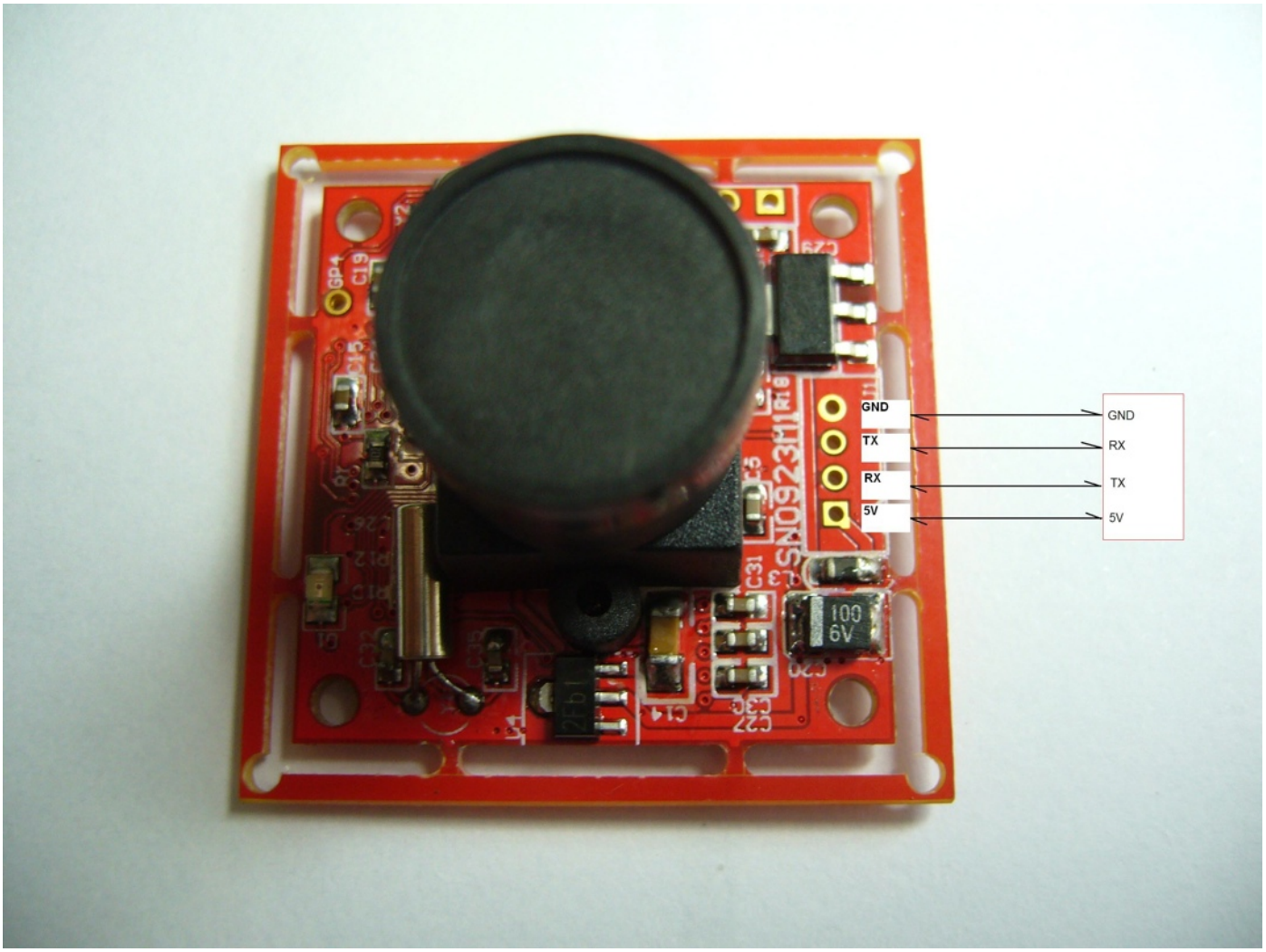
RSV5-C328 camera is controlled by twelve commands, like *Sync, Initial, Snapshot, GetPicture* and so on. Below is a complete list of camera commands. For complete list of camera commands and their parameters, Please read RSV5-C328 user manual in details. These commands are represented by two bytes and has four arguments. The complete command packet is always six bytes long.

Picture below contains full set of camera commands. each commands is coded in your code such that the command itself (command ID) in the second column which is two bytes, followed by four parameters each of them is one byte. The total is six bytes for each command

Command	ID Number	Parameter1	Parameter2	Parameter3	Parameter4
Initial	AA01h	00h	Color Type	RAW Resolution (Still image only)	JPEG Resolution
Get Picture	AA04h	Picture Type	00h	00h	00h
Snapshot	AA05h	Snapshot Type	Skip Frame Low Byte	Skip Frame High Byte	00h
Set Package Size	AA06h	08h	Package Size Low Byte	Package Size High Byte	00h
Set Baudrate	AA07h	1st Divider	2nd Divider	00h	00h
Reset	AA08h	Reset Type	00h	00h	xxh*
Power Off	AA09h	00h	00h	00h	00h
Data	AA0Ah	Data Type	Length Byte 0	Length Byte 1	Length Byte 2
SYNC	AA0Dh	00h	00h	00h	00h
ACK	AA0Eh	Command ID	ACK counter	00h / Package ID Byte 0	00h / Package ID Byte 1
NAK	AA0Fh	00h	NAK counter	Error Number	00h
Light Frequency	AA13h	Frequency Type	00h	00h	00h

* If the parameter is 0xFF, the command is a special Reset command and the firmware responds to it immediately.





Future Electronics Egypt Ltd. (Arduino Egypt).

Procedures of How to Use the Camera module for Snapshot:

1) The camera is connected is connected to pc serial port (for RS232 Model) or To Arduino / Micro-controller (for TTL Model) as shown in picture below.

2) The Camera is powered by 5V source through the 5v pin as shown in picture below.

3) Once the power applied to Camera, a red LED blinks one time as indication for power

4) **Synchronization Step**: SYNC command is sent from your PC 25-60 times until an ACK command is received from the camera . The hexadecimal form of the SYNC command is : (AA0D00000000h), where h denotes hexadecimal.

5) The PC or Arduino keep sending SYNC command and keep watching until receiving acknowledgment command from the camera. An example of the form of acknowledgment command is (AA 0E 01 00 00 00).

6) **Important Note:** a 10ms delay MUST be used between SYNC Commands in order to get the camera SYNC easily.

7) **Initialization Command:** after SYNC step, a command is sent by pc or Arduino to initialize and setup the camera. Initial(C328R.ColorType, C328R.PreviewResolution, C328R.JpegResolution). An example of the form of the command is (AA 01 00 07 07 07). This example translate into: initialize camera (AA 01) with Jpeg color (07) and with preview resolution of 640 x 480 (07) and JPEG resolution of 640 x 480 (07).

8) An acknowledgment is received from camera

9) **Send package size command:** This command is sent by PC or Arduino to set the size of the data package which is used to transmit the compressed JPEG image data from the camera to the PC or Arduino. This command should be issued before sending SNAPSHOT or GET PICTURE commands to the camera. An example of this command is : (AA 06 08 00 02 00), this command define package size of 512 byte.

10) An acknowledgment is received from camera.

Important Note: Before snapshoot command give camera some time delay (1 second).

11) **Take a snapshot command**: this command is sent by pc or Arduino to camera and cause the camera to take a snapshot (picture) and store it to camera buffer. An example of the form of this command is (AA 05 00 00 00 00).

12) An acknowledgment is received from camera.

13) **Get picture command:** The PC or Arduino issues this command to request a picture from the camera. This command is sent only one time, but data will come back multiple times - to extract out all the images....and it will come back in the form of "package" as we have defined the size earlier in "set package size". u have calculate the formula u will know when to stop asking for the next package. And there is a verification byte - u have to use that to make sure data is correct. By knowing the image size , we are able to calculate the amount of packages to be received, just like it is told in the user manual: "Number of packages = Image size / (Package size – 6)". because 6 bytes are used up for non-data purpose. They are reserved as follows: 2 bytes for ID, 2 bytes for data length, N bytes of data, 2 bytes of checksum.

14) The image size is sent by camera as part of the data command in the next step.

Important Note: After Get Picture command give camera some time delay (1 second) and jet pic command is the same

15)**Acknowledgment then Data command**: this command is issued by the camera whre data start to transfer from camera buffer to Arduino.

14)**To assemble the data from the package**, u have to remove the 4 byte header (at beginning of data packet) + 2 byte trailer (at the end of data packet) and concatenate all these from all the package sequentially one after another.

Tips:

1)How to assemble picture file from data packages:

To assemble the data from the package, u have to remove the 4 byte header (at beginning of dat package) + 2 byte trailer (at the end of data package) and concatenate all these from all the package sequentially one after another.

2) Video Mode:

Just do not send SNAPSHOT command, the camera is effectively running in video mode, you just send GET_PICTURE as quickly as you can.

3)"Set Package Size" command must be sent from host to CAM - before "SNAPSHOT" command, which capture the image from the camera into the CAM memory buffer.

4) How Camera works: The camera creates an image in the buffer. Now, if you send the snapshot command, camera builds the image in the buffer but doesn't send it. You can send Get Picture command (with snapshot picture option set) to get that picture from the buffer. The snapshot remains in the buffer until you take a new snapshot or new JPEG/Raw picture though Get

Picture command. The picture also remains in the memory buffer as long as you have power connected to it.
Once you have got the data you can view the image, which depends on your platform.

Writing Command in Arduino Example:

1) Sending Command to camera (Example initialization command):

```
//this is initian commandfor the cmd info,please refer to the
datasheet of serial camera
char cmdinit[] = {
  0xAA, 0x01, 0x00, 0x07, 0x03, 0x03};

for(int i =0;i < 6;i ++){
    Serial.print(cmdinit[i]);//initialize the camera
  }
```

2)Reading data sent by camera to Arduino:

```
while(Serial.available() < 7);
  while(Serial.available() > 0){
     data = Serial.read();
     Serial.print(data);
  }
  delay(50);
```

3)Setting Baude Rate:

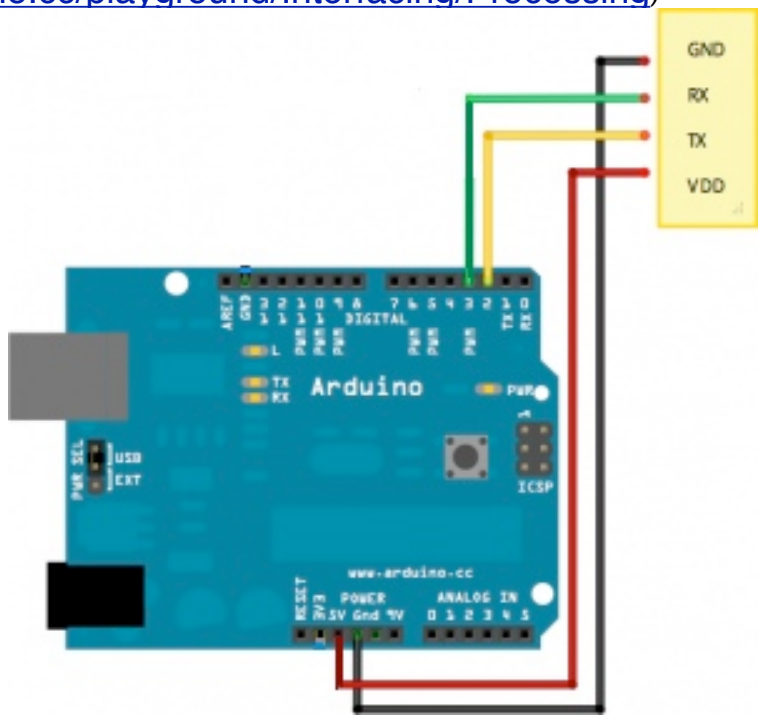
```
Serial.begin(115200)
```

Arduino Code:

The following code is used to work the camera using Arduino and get the picture on your pc using processing tool. As an alternative, you can save the data on SD card using Arduino card reader shield.

Processing is an open source language/ development tool for writing programs in other computers. Useful when you want those other computers to "talk" with an Arduino, for instance to display or save some data collected by the Arduino.

For examples about using processing with Arduino , click here (http://www.arduino.cc/playground/Interfacing/Processing)



Download required libraries:

1) Download C328 Library
2) Download Newsoftserial Libarary
3) Extract the zip files to "\arduino-0018\libraries\"
4) Start Arduino IDE, compile and download the following source code to Arduino:

Part 1) Arduino Code

```
#include <CameraC328R.h>
#include <NewSoftSerial.h>

#define LED_PIN 13
#define PAGE_SIZE 64
#define USB_BAUD 115200
#define CAMERA_BAUD 14400

NewSoftSerial mySerial(2, 3);
CameraC328R camera(&mySerial);

uint16_t pictureSizeCount = 0;

/**
 * This callback is called EVERY time a JPEG data packet is received.
 */
void getJPEGPicture_callback( uint16_t pictureSize, uint16_t packageSize,
uint16_t packageCount, byte* package )
{
  // packageSize is the size of the picture part of the package
  pictureSizeCount += packageSize;

  Serial.write(package,packageSize);

  if( pictureSizeCount >= pictureSize )
  {
    digitalWrite( LED_PIN, LOW );
    Serial.flush();
  }
}

void setup()
{
  Serial.begin( USB_BAUD );
  mySerial.begin(CAMERA_BAUD);

  pinMode( LED_PIN, OUTPUT );
  digitalWrite( LED_PIN, LOW );
}

void loop()
{
  if( Serial.available() ){

    while(Serial.read() != -1);

    digitalWrite( LED_PIN, HIGH );

    if( !camera.sync() )
    {
```

```
      Serial.println( "Sync failed." );
      return;
    }

    if( !camera.initial( CameraC328R::CT_JPEG, CameraC328R::PR_160x120,
CameraC328R::JR_640x480 ) )
    {
      Serial.println( "Initial failed." );
      return;
    }

    if( !camera.setPackageSize( 64 ) )
    {
      Serial.println( "Package size failed." );
      return;
    }

    if( !camera.setLightFrequency( CameraC328R::FT_50Hz ) )
    {
      Serial.println( "Light frequency failed." );
      return;
    }

    if( !camera.snapshot( CameraC328R::ST_COMPRESSED, 0 ) )
    {
      Serial.println( "Snapshot failed." );
      return;
    }

    pictureSizeCount = 0;
    if( !camera.getJPEGPicture( CameraC328R::PT_JPEG, PROCESS_DELAY,
&getJPEGPicture_callback ) )
    {
      Serial.println( "Get JPEG failed." );
      return;
    }
  }
}
```

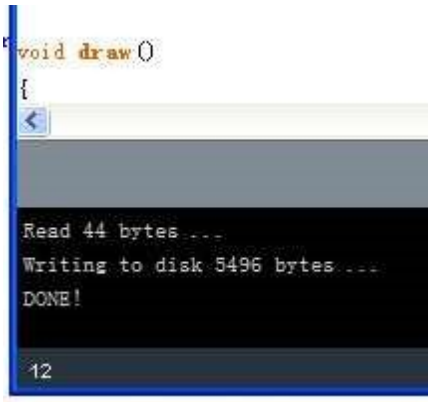
Start Processing and run the following code:

Part 2) Arduino and Processing

```
import processing.serial.*;

Serial myPort;
String filename = "photo.jpg";
byte[] photo = {
};
Boolean readData = false;
```

```
void setup()
{
  println( Serial.list() );
  myPort = new Serial( this, Serial.list()[0], 38400 );
}

void draw()
{
  byte[] buffer = new byte[64];
  if( readData )
  {
    while( myPort.available() > 0 )
    {
      int readBytes = myPort.readBytes( buffer );
      print( "Read " );
      print( readBytes );
      println( " bytes ..." );
      for( int i = 0; i < readBytes; i++ )
      {
        photo = append( photo, buffer[i] );
      }
    }
  }
  else
  {
    while( myPort.available() > 0 )
    {
      print( "COM Data: " );
      println( myPort.readString() );
    }
  }
}

void keyPressed()
{
  if( photo.length > 0 ) {
    readData = false;
    print( "Writing to disk " );
    print( photo.length );
    println( " bytes ..." );
    saveBytes( filename, photo );
    println( "DONE!" );
  }
  else {
    readData = true;
    myPort.clear();
    println( "Waiting for data ..." );
  }
}
```

Press the number corresponding to Arduino serial port on your PC. After a while, it will stop and you can press ANY key again, it will show you:



and you can see photo below and image file in your Processing project folder