Serial Camera (CMOS Serial Port (UART) Camera Module)

Technical Manual Rev 1r0

Capture JPEG images and send them via RS232 UART interface. This serial camera module makes it very easy for your microcontroller circuits to add image capture functions for various applications. VGA 640 x 480 image resolution, 115kbps UART rate. Compatible in all gizdDuino boards and MCUs.

Features:

- Color serial camera module with lens
- CMOS serial port (UART) camera module
- Resolution: VGA 640*480
- Image Format: JPEG
- Baud Rate: adjustable
- Image Sensor: 1/4” CMOS OV7725
- Signal System: PAL, NTSC

General Specifications:

Power Supply: 5V logic
Output: UART and CVBS
Baud Rate: 115200
Compressed Image Format: JPEG
Operating Temperature: -10°C to 60°C (RH90% Max)
Storage Temperature: -20°C to 70°C (RH90% Max)
Dimensions: 32mm*32mm,38mm*38mm

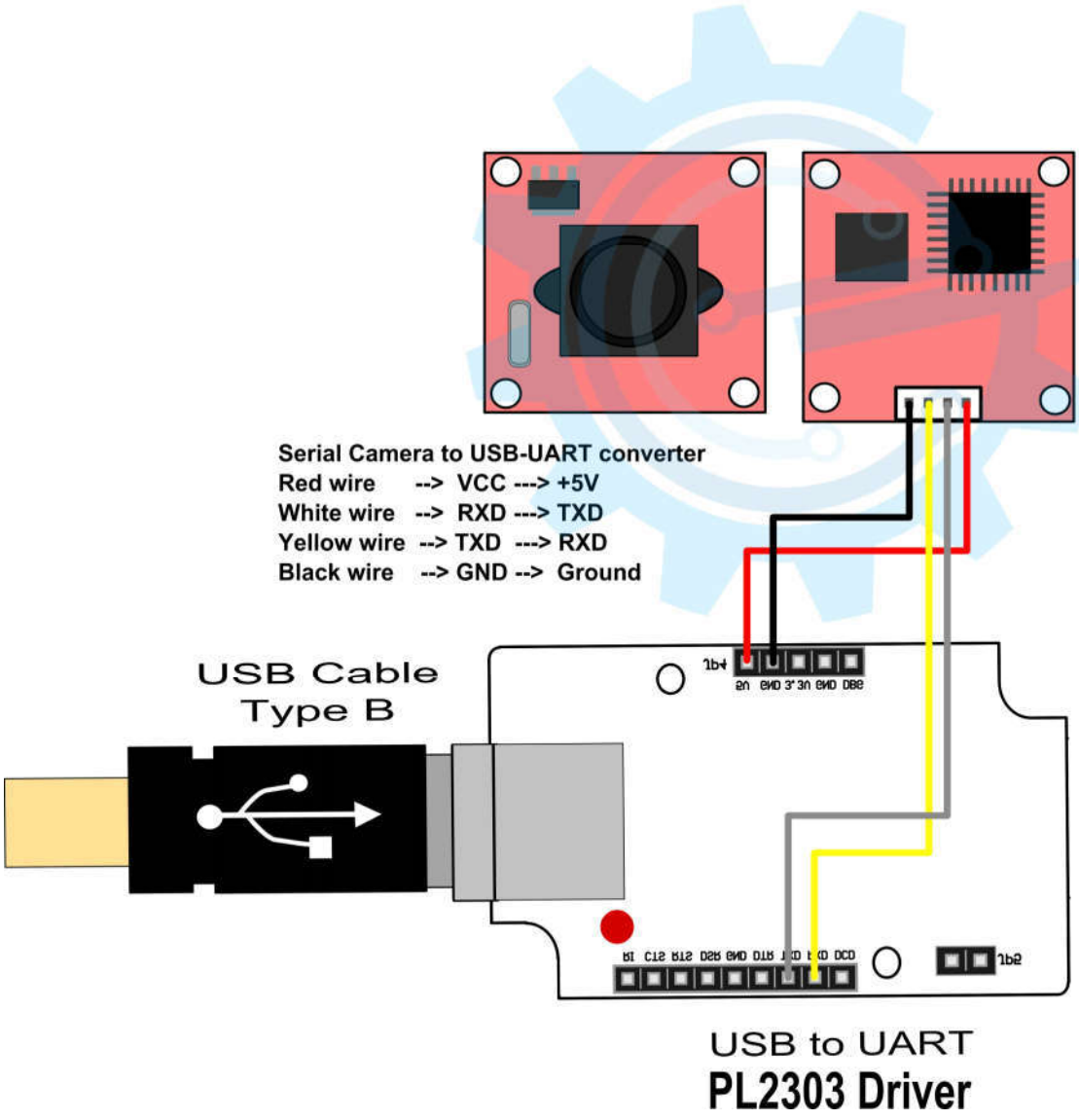


Figure 1: Using PL2303 driver

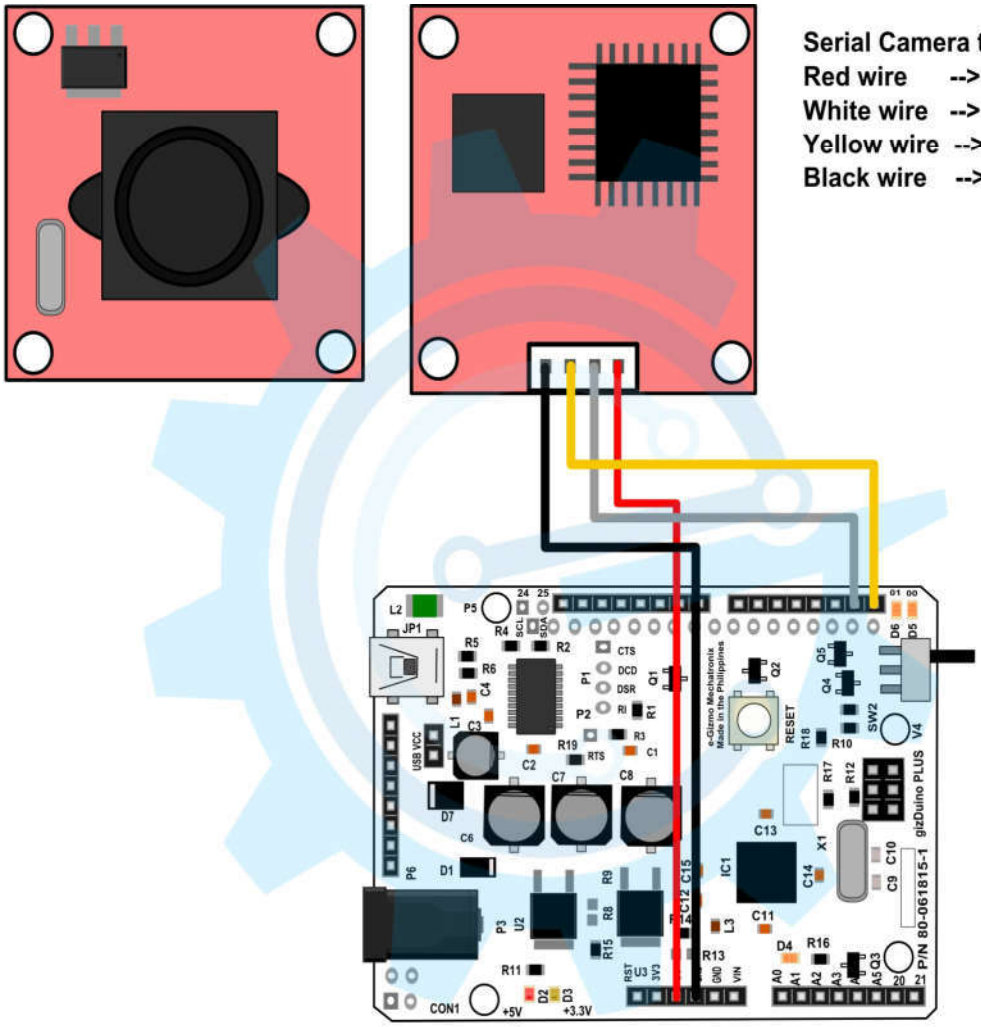


Figure 2: gizDuino PLUS ATmega644P

```
//  File SerialCamera_DemoCode_CJ-OV528.ino
//  8/8/2013 Jack Shao
//  Rewritten by: E-gizmo Mechatronix Central
//  Demo code for using seeeduino or Arduino board to cature jpg format
//  picture from seeed serial camera and save it into sd card. Push the
//  button to take the a picture .
//  For more details about the product please check http://www.seeedstudio.com/depot/

#include <SPI.h>
#include <arduino.h>
#include <SD.h>

#define PIC_PKT_LEN     128                  //data length of each read, dont set this too big
because ram is limited
#define PIC_FMT_VGA    7
#define PIC_FMT_CIF    5
#define PIC_FMT_OCIF   3
#define CAM_ADDR       0
#define CAM_SERIAL     Serial

#define PIC_FMT        PIC_FMT_VGA

File myFile;

const byte cameraAddr = (CAM_ADDR << 5);  // addr
const int buttonPin = 19;                 // the number of the pushbutton pin
unsigned long picTotalLen = 0;            // picture length
int picNameNum = 0;

/*************************************************************************/
void setup()
{
  Serial.begin(115200);
  pinMode(buttonPin, INPUT);    // initialize the pushbutton pin as an input
  digitalWrite(buttonPin,HIGH);
  Serial.println("Initializing SD card....");
  pinMode(10,OUTPUT);          // CS pin of SD Card Shield
  pinMode(8,OUTPUT);
  pinMode(7,OUTPUT);
  pinMode(6,OUTPUT);

  if (!SD.begin(10))
  {
    Serial.print("initialization failed");
    return;
  }
  Serial.println("initialization done.");
  digitalWrite(8, HIGH);
  initialize();
}
/*************************************************************************/
```

```
void loop()
{
  int n = 0;
  while(1){
    Serial.println("\r\nPress the button to take a picture");
    while (digitalRead(buttonPin) == HIGH);      //wait for buttonPin status to HIGH
    if(digitalRead(buttonPin) == LOW){
      delay(50);                          //Debounce
      if (digitalRead(buttonPin) == LOW)
      {
        Serial.println("\r\nCOPYING JPG TO CARD, PLS WAIT...");
        delay(200);
        if (n == 0) preCapture();
        Capture();
        GetData();
      }
      Serial.print("\r\nTaking pictures success ,number : ");
      Serial.println(n);
      n++ ;
    }
  }
}
/*********************************************************************/
void clearRxBuf()
{
  while (Serial.available())
  {
    Serial.read();
  }
}
/*********************************************************************/
void sendCmd(char cmd[], int cmd_len)
{
  for (char i = 0; i < cmd_len; i++) Serial.print(cmd[i]);
}
/*********************************************************************/
void initialize()
{
  char cmd[] = {
    0xaa,0x0d|cameraAddr,0x00,0x00,0x00,0x00  }
  ;
  unsigned char resp[6];

  Serial.setTimeout(500);
  while (1)
  {
    //clearRxBuf();
    sendCmd(cmd,6);
    if (Serial.readBytes((char *)resp, 6) != 6)
    {
      continue;
```

```
    }
    if (resp[0] == 0xaa && resp[1] == (0x0e | cameraAddr) && resp[2] == 0x0d && resp[4] ==
0 && resp[5] == 0)
    {
      if (Serial.readBytes((char *)resp, 6) != 6) continue;
      if (resp[0] == 0xaa && resp[1] == (0x0d | cameraAddr) && resp[2] == 0 && resp[3] == 0
&& resp[4] == 0 && resp[5] == 0) break;
    }
  }
  cmd[1] = 0x0e | cameraAddr;
  cmd[2] = 0x0d;
  sendCmd(cmd, 6);
  Serial.println("\nCamera Ready.");
  digitalWrite(6, HIGH);
  digitalWrite(7, HIGH);
}
/*************************************************************************/
void preCapture()
{
  char cmd[] = {
    0xaa, 0x01 | cameraAddr, 0x00, 0x07, 0x00, PIC_FMT   };
  unsigned char resp[6];

  Serial.setTimeout(100);
  while (1)
  {
    clearRxBuf();
    sendCmd(cmd, 6);
    if (Serial.readBytes((char *)resp, 6) != 6) continue;
    if (resp[0] == 0xaa && resp[1] == (0x0e | cameraAddr) && resp[2] == 0x01 && resp[4] ==
0 && resp[5] == 0) break;
  }
}
void Capture()
{
  char cmd[] = {
    0xaa, 0x06 | cameraAddr, 0x08, PIC_PKT_LEN & 0xff, (PIC_PKT_LEN>>8) & 0xff ,0  };
  unsigned char resp[6];

  Serial.setTimeout(100);
  while (1)
  {
    clearRxBuf();
    sendCmd(cmd, 6);
    if (Serial.readBytes((char *)resp, 6) != 6) continue;
    if (resp[0] == 0xaa && resp[1] == (0x0e | cameraAddr) && resp[2] == 0x06 && resp[4] ==
0 && resp[5] == 0) break;
  }
  cmd[1] = 0x05 | cameraAddr;
  cmd[2] = 0;
  cmd[3] = 0;
```

```
  cmd[4] = 0;
  cmd[5] = 0;
  while (1)
  {
    clearRxBuf();
    sendCmd(cmd, 6);
    if (Serial.readBytes((char *)resp, 6) != 6) continue;
     if (resp[0] == 0xaa && resp[1] == (0x0e | cameraAddr) && resp[2] == 0x05 && resp[4] ==
0 && resp[5] == 0) break;
  }
  cmd[1] = 0x04 | cameraAddr;
  cmd[2] = 0x1;
  while (1)
  {
    clearRxBuf();
    sendCmd(cmd, 6);
    if (Serial.readBytes((char *)resp, 6) != 6) continue;
     if (resp[0] == 0xaa && resp[1] == (0x0e | cameraAddr) && resp[2] == 0x04 && resp[4] ==
0 && resp[5] == 0)
     {
       Serial.setTimeout(1000);
       if (Serial.readBytes((char *)resp, 6) != 6)
       {
         continue;
       }
       if (resp[0] == 0xaa && resp[1] == (0x0a | cameraAddr) && resp[2] == 0x01)
       {
         picTotalLen = (resp[3]) | (resp[4] << 8) | (resp[5] << 16);
         Serial.print("picTotalLen:");
         Serial.println(picTotalLen);
         break;
       }
     }
  }

}
/************************************************************************/
void GetData()
{
  unsigned int pktCnt = (picTotalLen) / (PIC_PKT_LEN - 6);
  if ((picTotalLen % (PIC_PKT_LEN-6)) != 0) pktCnt += 1;

  char cmd[] = {
    0xaa, 0x0e | cameraAddr, 0x00, 0x00, 0x00, 0x00   };
  unsigned char pkt[PIC_PKT_LEN];

  char picName[] = "pic00.jpg";
  picName[3] = picNameNum/10 + '0';
  picName[4] = picNameNum%10 + '0';

  if (SD.exists(picName))
```

```
  {
    SD.remove(picName);
  }

  myFile = SD.open(picName, FILE_WRITE);
  if(!myFile){
    Serial.println("myFile open fail...");
  }
  else{
    Serial.setTimeout(1000);
    for (unsigned int i = 0; i < pktCnt; i++)
    {
      cmd[4] = i & 0xff;
      cmd[5] = (i >> 8) & 0xff;

      int retry_cnt = 0;
retry:
      delay(10);
      clearRxBuf();
      sendCmd(cmd, 6);
      uint16_t cnt = Serial.readBytes((char *)pkt, PIC_PKT_LEN);

      unsigned char sum = 0;
      for (int y = 0; y < cnt - 2; y++)
      {
        sum += pkt[y];
      }
      if (sum != pkt[cnt-2])
      {
        if (++retry_cnt < 100) goto retry;
        else break;
      }

      myFile.write((const uint8_t *)&pkt[4], cnt-6);
      //if (cnt != PIC_PKT_LEN) break;
    }
    cmd[4] = 0xf0;
    cmd[5] = 0xf0;
    sendCmd(cmd, 6);
  }
  myFile.close();
  picNameNum ++;
}
```

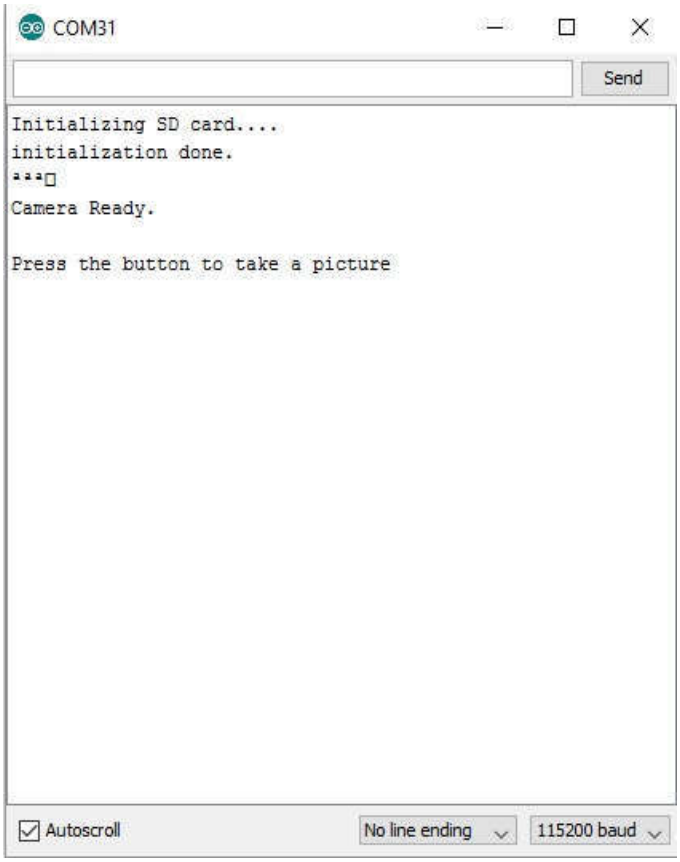
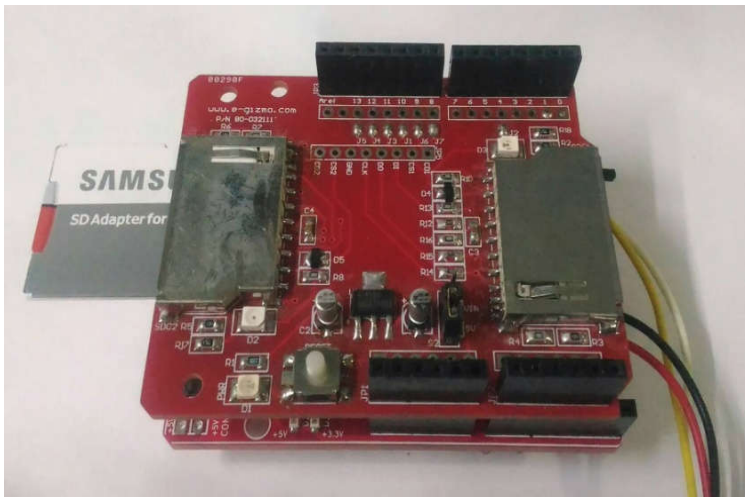
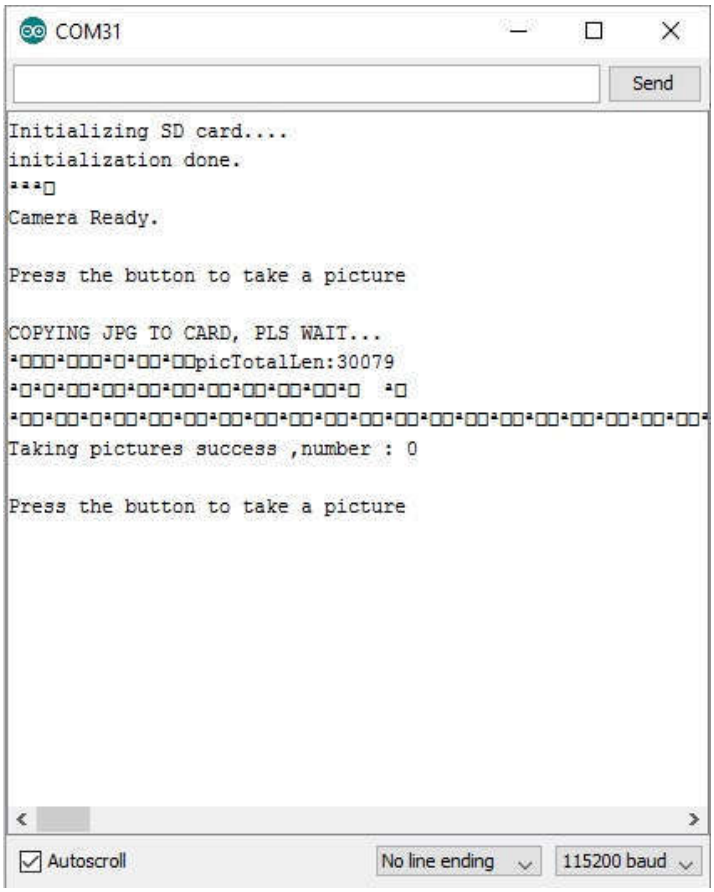


Figure 3: Serial Monitor initializing SD card and Camera ready.



Using SD/MMC Card shield you can save the image taken from the Serial Camera. Just place the SD/MMC card to the SDC2 SD card Slot. Make sure your sd card is formatted.

The filename starts with PIC00.JPG, PIC01.JPG and So on, if you take a pictures continously. But if you reset the power or press the reset button and take picture again it will overwritten the PIC00.JPG image.



By pressing the button or put a ground into D19/A5 of gizDuino. It will start COPYING JPG TO CARD, wait until the ***Taking pictures success!***

Figure 4: Serial Monitor while pressing the button once until its done.

Sample Image.