WS2812 LED Drive for ATMega8,128,162,328P

(*anyone who can help me in this editor and English, I appreciate and also many thanks)

A copy of this doc is in sub Folder Doc of WS2812_Drive_V2 project)

Development

This Drive as Developed in C++ using Atmel Studio 7.0 and work very well and it was tested with Atmel ATMega8, ATMega128, ATMega162 and ATMega328P(same as Arduino Uno Microcontroller) using a crystal frequency of 16Mhz.

Purpose

The purpose this Drive is to control WS2812 LED send already formatted data string.This Drive work with LED matrix with size LxC where L is number of Rows and C is number of Columns of LED, allowing size of 1x1,2x1,8x1, 2x2,4x4,8x8, 16x16 and so on. The unique limitation of Drive is a memory available in microcontroller. The Drive function responsible for communication of LED matrix was made totally in assembler language because a reason of speed, this version works only with a crystal frequency of 16Mhz and in the future version i implement instance of this Drive for control of multiples WS2812 LED and also with the option to organizer panels in cascade or separate.

Graphics

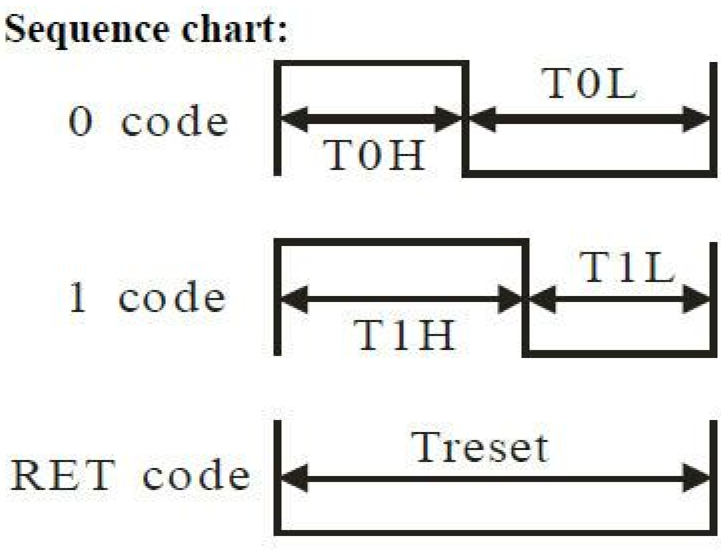
Added with this Drive has a Graphic engine with capacity to Draw many shapes and Methods like Lines,Boxes, Circles, Scrolls and Text, allowing programmers to make easily any App for a Panel or Panels of LED matrix.

Author : **João D’Artagnan Antunes Oliveira**

Brasília, Brazil, 31 November 2015

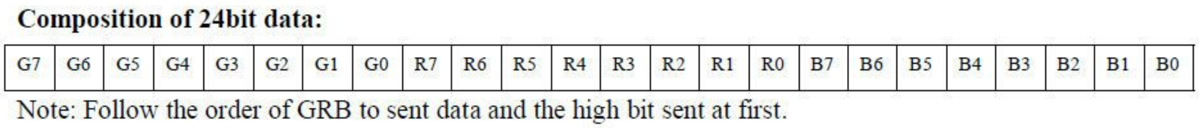
WS2812 LED

WS2812 is a intelligent control LED light source that the control circuit and RGB chip are integrated in a package of 5050 components. It internal include intelligent digital port data latch and signal reshaping amplification drive circuit. Also include a precision internal oscillator and a 12V voltage programmable constant current control part, effectively ensuring the pixel point light color height consistent.The data transfer protocol use single NZR communication mode. After the pixel power-on reset, the DIN port receive data from controller, the first pixel collect initial 24bit data then sent to the internal data latch,the other data which reshaping by the internal signal reshaping amplification circuit sent to the next cascade pixel through the DO port. After transmission for each pixel. the signal to reduce 24bit. pixel adopt auto reshaping transmit technology, making the pixel cascade number is not limited the signal transmission, only depend on the speed of signal transmission. Below a description of Timing chart and respective values and other technical references.



Data transfer time(TH+TL=1.25µs±600ns)

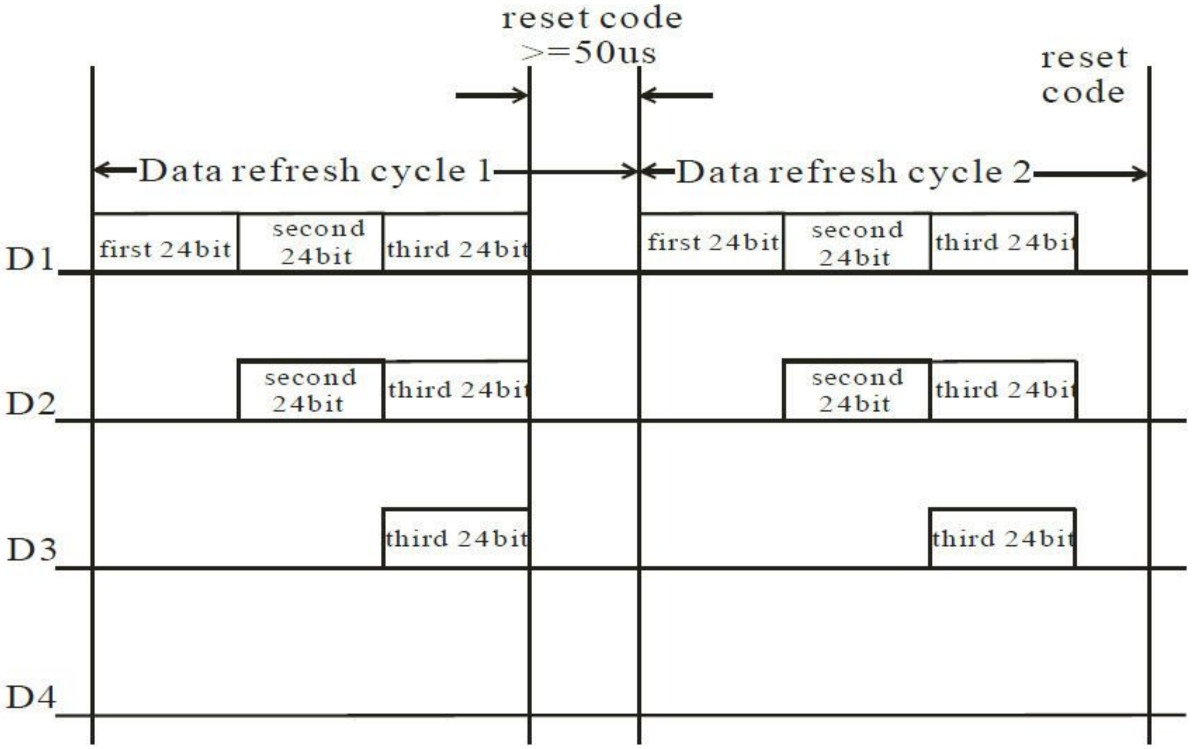
- **T0H 0 code ,high voltage time 0.35us ±150ns**
- **T1H 1 code ,high voltage time 0.7us ±150ns**
- **T0L 0 code , low voltage time 0.8us ±150ns**
- **T1L 1 code ,low voltage time 0.6us ±150ns**
- **RES low voltage time Above 50µs**



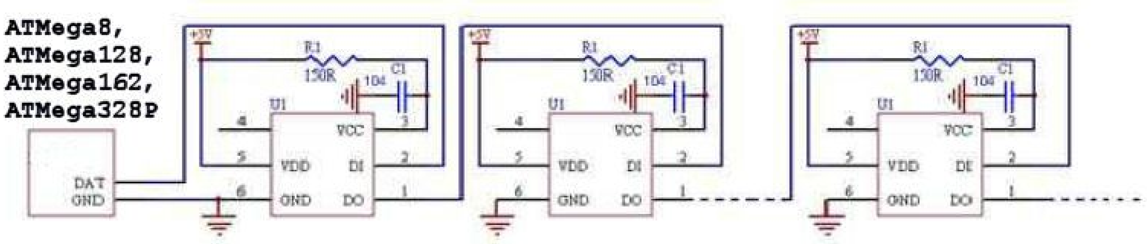
Composition of 24bit data:

G7	G6	G5	G4	G3	G2	G1	G0	R7	R6	R5	R4	R3	R2	R1	R0	B7	B6	B5	B4	B3	B2	B1	B0

Note: Follow the order of GRB to sent data and the high bit sent at first.



Typical application circuit:

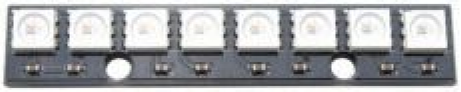


One thing i observed during development is that data tx line must be remainder low during all time, if left this line High instead Low an error occur during bit transmission. The main transmission function send 24 bits of green,red,blue values of each ws2812 LED and next transmission must be occur less than a reset time 50us, the Drive take this in consideration and keep this time between transmission a short as possible, generally an average of 2us.

Some pictures examples of WS2812 LED panel matrix.

A single dot WS2812 LED

Cluster of WS2812 LED



1x8 WS2812 LED Panel

4x4 WS2812 LED Panel

8x8 WS2812 panel , this was used for development.

Many circular WS2812 Panel, we must consider a Panel with 1(row)x(n)LED(column) when pass parameters for the drive.

Source Code

This drive had developed complete in C++ language using Atmel Studio 7.0 and some part in assembler language of AVR microcontrollers for reason of speed. Below description of implemented function of each class object used.

Class class_Time

Function	Description
void PrepareForSleep(void)	Prepare Timer for microprocessor enter to Sleep state
void RestoreFromSleep(void)	Restore Timer from Sleep state
void Delay7Nops(uint16_t count)	Delay 7 NOPS instruction a count times
void DelayMsS(uint16_t timeMs)	Delay timeMS microseconds by software
uint32_t GetMilliseconds(void)	Get elapsed milliseconds from CPU reset
void SetElapsed(uint32_t * timeVariable)	Set Elapsed Time in milliseconds to timeVariable
uint32_t GetElapsed(uint32_t *timeVariable)	Get Elapset time in milliseconds on timeVariable since last SetElapsed time
void DelayMsI(uint32_t timeMs);	Delay timeMS milliseconds using interrupt timer, global interrupts must be enabled i.e. sei()
void ChronoMark(void);	Mark an actual time
uint32_t ChronoNow(void)	Get Elapsed time since ChronoMark
void SetTimeClockCallback(void (*TimePumpCallback)(uint8_t hour,uint8_t minute,uint8_t second))	Define callback routine for process time, this function when enabled is called 10 times per second
void SetTimeClockCallback(void (*TimePumpCallback)(uint8_t hour,uint8_t minute,uint8_t second))	Define callback routine for a timer interrupt, this function when Enabled is called 1250 times per second
void SetTimeInterruptCallback(void (*TimeInterruptCallback)(void))	Define callback routine for a timer interrupt, this function when Enabled is called 1250 times per second
void SetClockState(bool inUse)	Enable clock call back function
void SetInterruptState(bool inUse)	Enable Interrupt State call back function

void GetTime(uint8_t &hour,uint8_t &minute,uint8_t &second);	Get Current Time hour,minute,second
void SetTime(uint8_t hour,uint8_t minute,uint8_t second);	Set Current Time hour,minute,second

Class Color	
Function	**Description**
Color(void) Constructor	Initialize Color with Black
Color(uint8_t red,uint8_t green,uint8_t blue) Constructor	Initialize Color with red,green,blue values
Color FromRGB(uint8_t red,uint8_t green,uint8_t blue)	Define Color from red,green,blue values
bool operator== (Color color)	Used for Color comparison

Class class_GraphBasic	
Function	**Description**
class_GraphBasic() Constructor	Initialize Graph with panel size of 1x1 LED
class_GraphBasic(int rows,int cols)	Initialize Graph with panel size of number of Rows and Cols
uint8_t GetRows(void)	Get number of rows of panel
uint8_t GetCols(void)	Get number of columns of panel
int inline MinX(void)	Get Minimal value of X coordinate of panel
int inline MaxX(void)	Get Maximum value of X coordinate of panel
int inline MinY(void)	Get Minimal value of Y coordinate of panel

int inline MaxY(void)	Get Maximum value of Y Coordinate of panel
void SetBackColor(Color color)	Set Background Color of panel used by some function like Cls(),PutC,PutS,PutSF etc.
void SetBackColor(uint8_t red,uint8_t green,uint8_t blue)	Set Background Color of panel using red,green,blue values
Color GetBackColor(void)	Get Background Color of panel
void SetForeColor(Color color)	Set Fore Color of panel using Color class
void SetForeColor(uint8_t red,uint8_t green,uint8_t blue);	Set Fore Color of panel using red,green, values
void SetPixel(int x,int y)	Set Pixel LED on panel at x,y coordinates using actual Forecolor
void SetPixel(int x,int y,Color pixelColor)	Set Pixel LED on panel using Color class at coordinates x,y
void SetPixel(int x,int y,uint8_t red,uint8_t green,uint8_t blue)	Set Pixel LED on panel using red,green,blue values
Color GetPixel(int x,int y)	Get Pixel Color of panel at coordinates x,y
void VertLine(int x,int yi,int yf)	Draw a Vertical Line on panel x coord and yi=start y coord and yf=final y coord using actual forecolor
void VertLine(int x,int yi,int yf,Color pixelColor)	Draw a Vertical Line on panel x coord and yi=start y coord and yf=final y coord using Color
void VertLine(int x,int yi,int yf,uint8_t red,uint8_t green,uint8_t blue)	Draw a Vertical Line on panel x coord and yi=start y coord and yf=final y coord using red,green,blue values
void HorzLine(int x,int yi,int yf)	Draw a Horizontal Line on panel x coord and yi=start y coord and yf=final y coord using actual forecolor
void HorzLine(int x,int yi,int yf,Color pixelColor)	Draw a Horizontal Line on panel x coord and yi=start y coord and yf=final y coord using Color
void HorzLine(int x,int yi,int yf,uint8_t red,uint8_t green,uint8_t blue)	Draw a Horizontal Line on panel x coord and yi=start y coord and yf=final y coord using red,green,blue values
void Box(int xi,int yi,int xf,int yf)	Draw a Box on panel starting at coods xi,yi ending at coords xf,yf using Forecolor
void Box(int xi,int yi,int xf,int yf,Color pixelColor)	Draw a Box on panel starting at coods xi,yi ending at coords xf,yf using Color

void Box(int xi,int yi,int xf,int yf,uint8_t red,uint8_t green,uint8_t blue)	Draw a Box on panel starting at coods xi,yi ending at coords xf,yf using red,green,blue values
void BoxFull(int xi,int yi,int xf,int yf)	Draw a Box Full on panel starting at coods xi,yi ending at coords xf,yf using Forecolor
void BoxFull(int xi,int yi,int xf,int yf,Color pixelColor)	Draw a Box Full on panel starting at coods xi,yi ending at coords xf,yf using Color
void BoxFull(int xi,int yi,int xf,int yf,uint8_t red,uint8_t green,uint8_t blue)	Draw a Box Full on panel starting at coods xi,yi ending at coords xf,yf using red,green,blue values
void Line(int xi,int yi,int xf,int yf)	Draw a Line on panel starting at coods xi,yi ending at coords xf,yf using Forecolor
void Line(int xi,int yi,int xf,int yf,Color pixelColor)	Draw a Line on panel starting at coods xi,yi ending at coords xf,yf using Color
void Line(int xi,int yi,int xf,int yf,uint8_t red,uint8_t green,uint8_t blue)	Draw a Line on panel starting at coods xi,yi ending at coords xf,yf using red,green,blue values
void Circle(int x,int y,int radius)	Draw a Circle on panel at x,y coordinates with radius using Forecolor
void Box(int x,int y,int radius,Color pixelColor)	Draw a Circle on panel at x,y coordinates with radius using Color
void Box(int x,int y,int radius,uint8_t red,uint8_t green,uint8_t blue)	Draw a Circle on panel at x,y coordinates with radius using red,green,blue values
void Cls(void)	Clear LED on panel using Background color
void RotateX(int xi,int xf,int yi,int yf,bool shiftOnly=false)	RotateX a region of LED panel starting at xi,yi coordinates and ending at xf,yf coordinates on X axis, if xi<xf Rotation is LEFT else RIGHT if shiftOnly=false only shift is performed
void RotateY(int xi,int xf,int yi,int yf,bool shiftOnly=false)	RotateY a region of LED panel starting at xi,yi coordinates and ending at xf,yf coordinates on Y axis, if yi<yf Rotation is UP else DOWN if shiftOnly=false only shift is performed
void Locate(int x,int y)	Set Cursor Location for future use in Text functions
void PutC(char c,bool useBackColor=false)	Draw a Character on Screen at previous coordinates defined by Locate function if useBackColor=true Text off Characters if drawing using Backcolor Color
void PutS(const char *c,bool useBackColor=false)	Draw a String of Character on Screen at previous coordinates defined by Locate function if useBackColor=true Text off Characters if drawing using Backcolor Color

void PutSF(const char *c,bool useBackColor=false)	Draw a String of Character Placed into Flash Memory on Screen at previous coordinates defined by Locate function if useBackColor=true Text off Characters if drawing using Backcolor Color
uint8_t inline GetTextWidth(void)	Get Text Width in Pixels
uint8_t inline GetTextHeight(void)	Get Text Height in Pixels

Class_WS2812_Drive

Function	**Description**
class_WS2812_Drive()	Initialize WS2812 Drive with panel size of 1x1 LED
class_WS2812_Drive(int rows,int cols)	Initialize WS2812 Drive with size of number of Rows and Cols

Functions **Reset** and **WriteRGBToDevice** to perform respective order;

- **Reset** Generate a Reset time for WS2812 LED
- **WriteRGBToDevice** send a string of 24 bits to WS2812 LED versus number of Rows and Cols defined by class_WS2812

These functions is called by and manipulated by GraphBasic class, i only mentioned here for further implementations but in normal use avoid to used this functions.

DEMO

A DEMO version was created with below examples.

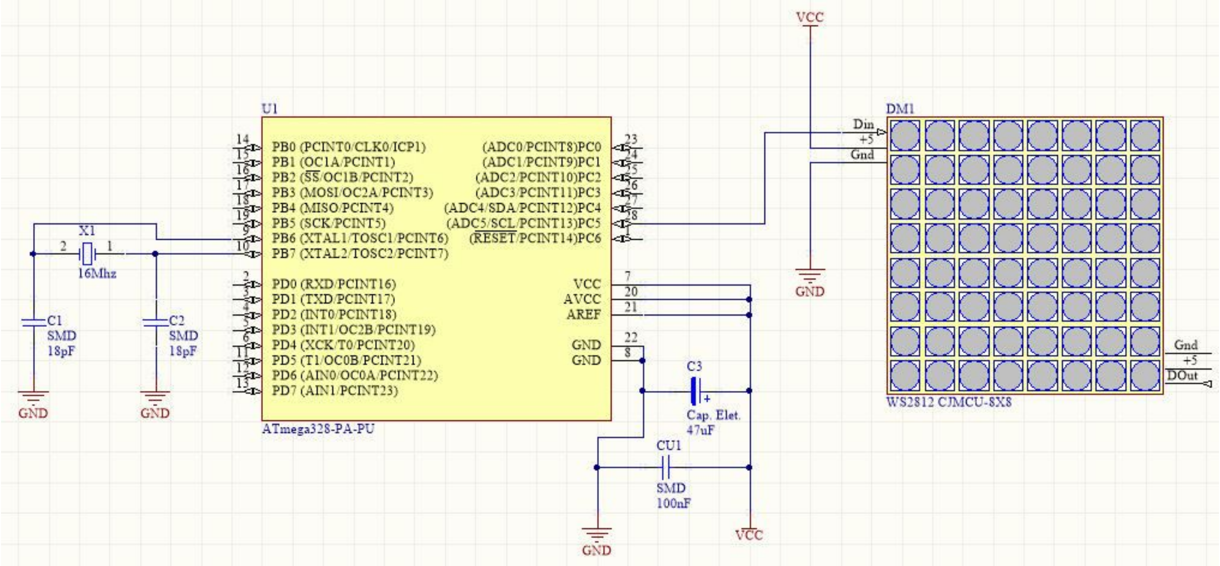
- **DrawText**
  - Draw a Text “Drive Version 1.02” Scrolling a Text from Right to Left
- **DrawVertLines**
  - Draw 8 Colored Vertical Lines .
- **DrawHorzLines**
  - Draw 8 Colored Horizontal Lines.
- **DrawBoxes**
  - Draw an animation of Boxes changing size and color.
- **DrawLines**
  - Draw an animation of Lines starting at a top (0,0 ending (7,7) and moviment pixel by pixel
- **DrawCircles**
  - Draw randomic Circle at random position and randomic Colors
- **RotateX**
  - Rotate a Sinusoidal Bar With 4 differents colors intercalated rotations LEFT AND RIGHT
- **Fades**
  - Full fades starting a Blue to Write at entire panel screen.
- **DrawBoxesFull**
  - Draw a Animation of Boxes Full Colored
- **RotatesXY**
  - Perform a concentric rotation of variable Color intensity and last a randomic concentric drawing of pixels.

After this, the DEMO start to DrawText and keep loop forever.

Click Here do See This Video Demo

Schematic

In Sub Folder Doc of WS2812_Drive_V2 have a doc Generic_Board.pdf with design of board projected by me to develop own projects, actually this a pin used is **PORTC PIN 0** named
EXPADC0-RW. This board have a FTDI232 Chip for USB connection and some generic pins. This schematic is free.



Wiring Schematic

Compiler Settings

A Following compiler parameters is used in this project

SYMBOL **-DF_CPU=16000000L -D_TIME_TIMER_NUMBER_=0**
OPTIMIZATION **-OS**
MISCELLANEOUS **-std=gnu++11**

In _GraphBasic.H define **_GRAPH_BIG_CHAR_SIZE_** To use character font 5x8 or **_GRAPH_NORMAL_CHAR_SIZE** To use character font 3x5

Disclaimers

Author: João D´Artagnan Antunes Oliveira

The Redistribution and use in source code forms and Electronic Schematic Diagram with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE OR HARDWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.