Ticket Display System with Arduino ⊕ HDMI Shield - Part I

Introduction

Ticket Display System is a common device you see it everywhere. From supermarkets, groceries, clinics, museums, whenever people need to queue for services such ticket display system shall exist. Usually it is designed with a personal computer (PC) or single board computer (SBC) running Windows. Queue management software is preloaded in hard-drive and launched from PC startup.

Photo below shows similar system from a local supermarket.



But there are always alternatives to PC with less power consumption, shorter boot up time, and much less complexity. In this document I am going to show you how a simple ticket display system is finished with a HDMI Shield-Arduino combo. Few lines of code in Arduino IDE is enough to do it.

Materials

The following materials are required to run the example in this document.

• HDMI Shield version 2/1 (mandatory)



• STM32F103 driver board (mandatory)



• A 1080p compatible HDTV/Monitor with HDMI/DVI input (mandatory)



• HDMI cable & a MicroUSB cable (both mandatory)



• Jumper cables(optional)



• DUE Zipper board with ESP8266 as MCU host or other Arduino boards we have tested so far (Arduino DUE as an optional item in this document)





Working principle

is illustrated with annotations below:

(2) Host MCU initializes RA8876 and CH7035 HDMI encoder, display a background image, and enter into an infinite loop to wait for incoming serial message until '\n' new line character is received



Procedures

1. First we need to prepare the background image and digits in bitmap format. Digits from '0'-'9' and two files "background_1.bmpⁱ" and "background_2.bmpⁱⁱ" have been prepared for you. They are included in the \assets folder. Digits in bitmap format with naming convention char00xx.bmp denote ticket number with xx=hex code of digit. Files with _mask suffixes are used for sprite version which will be explained in Part II of this document.



2. Now we need to use an Image Converter software to convert BMP/JPEG to binary. It is available from RAiO's web site at <u>http://www.raio.com.tw/en/Support_RA887677.html</u>. Its user guide is written in Chinese but we will walk through its operation in this document.



3. Download and install. Launch it with pull-down menu options as screen shot below. There are only three pull-down menus and one checkbox to use.

Please notice that our RA8876 firmware works with RGB5:6:5 only.

You also need to have Microsoft Excel installed in your PC for this tool to work because a .xls file will be generated to show the address map of binary files created.



Now click the ${\bf Load}\ {\bf Picture}$ button, and browse to bitmap files one-by-one from **\assets** folder in the order

- char0020.bmp > char0030.bmp > ... > char0039.bmp
- char0020_mask.bmp > char0030_mask.bmp ...> char0039_mask.bmp
- background_1.bmp > background_2.bmp.

The order is important because bitmap files are converted to binary data with addresses in the same order of a structure declared in the Arduino Sketch (Ra8876_ticketdisplay_part1.ino).

When more images are added to the RAiO Image Tool you will see the byte size become larger. At the end we are consuming 4,712,832 bytes for the ticket display system. We have no problem with this consumption because the Serial Flash onboard of our HDMI Shield is a 128Mbit model which is equivalent to a 16 Mbytes data capacity.

t the end you will RAiO Image Tool v1.1.0.1	see user		eriace				
NAO IIIlage 1001 VI.1.0.1		e list view					
BMP PNG	Picture		File Name	Width	Height	Data Format	Size (Bytes) or Array
		1	char0020.b	108	216	16bpp mode (RGB 5:6:5)	46656
	Ο	2	char0030.b	108	216	16bpp mode (RGB 5:6:5)	46656
	1	3	char0031.b	108	216	16bpp mode (RGB 5:6:5)	46656
A set	2	4	char0032.b	108	216	16bpp mode (RGB 5:6:5)	46656
	3	5	char0033.b	108	216	16bpp mode (RGB 5:6:5)	46656
	4	6	char0034.b	108	216	16bpp mode (RGB 5:6:5)	46656
	5	7	char0035.b	108	216	16bpp mode (RGB 5:6:5)	46656
	6	8	char0036.b	108	216	16bpp mode (RGB 5:6:5)	46656
RA8871/RA8873/RA8876/RA8877		0	char0027 h	100	516	16hnn mode (PCP E+6+E)	ACCEC
Binary File (*.bin)	Picture		File Name	Width	Height	Data Format	Size (Bytes) or Array
16bpp mode (RGB 5:6:5)	7	9	char0037.b		216	16bpp mode (RGB 5:6:5)	46656
Load Picture Merging multiple binary f	iles 8	10	char0038.b	108	216	16bpp mode (RGB 5:6:5)	46656
Convert _	9	11	char0039.b	108	216	16bpp mode (RGB 5:6:5)	46656
		12	char0020	108	216	16bpp mode (RGB 5:6:5)	46656
	O	13	char0030	108	216	16bpp mode (RGB 5:6:5)	46656
	1	14	char0031	108	216	16bpp mode (RGB 5:6:5)	46656
	2	15	char0032	108	216	16bpp mode (RGB 5:6:5)	46656
	Picture		File Name	Width	Height	Data Format	Size (Bytes) or Array
	4	17	char0034	108	216	16bpp mode (RGB 5:6:5)	46656
	5	18	char0035	108	216	16bpp mode (RGB 5:6:5)	46656
	6	19	char0036	108	216	16bpp mode (RGB 5:6:5)	46656
	7	20	char0037	108	216	16bpp mode (RGB 5:6:5)	46656
	8	21	char0038	108	216	16bpp mode (RGB 5:6:5)	46656
	9	22	char0039	108	216	16bpp mode (RGB 5:6:5)	46656
		23	background	1280	720	16bpp mode (RGB 5:6:5)	1843200
		24	background	1280	720	16bpp mode (RGB 5:6:5)	1843200
	1						Þ

At the end you will see user interface like this :

- 4. Click **Convert**. Inside the **/assets** folder you will see three files generated. They are
 - All_76_Pic_65K.bin (binary file to be preloaded to Serial Flash)
 - All_76_Pic_65K.h (a brief guide in .h file, but we won't use it here)
 - All_76_Pic_65K.xls. (Excel file containing image parameters and addresses that we will use it in source code)

You may close the RAiO Image Tool now. Its task is finished.

5. Launch RA8876/77 AP Software. This program can be downloaded from http://www.raio.com.tw/en/Support_RA887677.html.

RAIO 瑞佑科技	About RAIO Products Technical Support
Software Tool Resource :	
AP Software tool User Guide 🕂	包繁體中文
AP Software tool Download 🕂	RA8876 77 AP v1.1.0.0.zip

By this time I have assumed the device driver for STM32F103 has been properly installed. Its procedure is described in the Getting Started Guideⁱⁱⁱ from page 17.

6. Now we need to connect the STM32F103 driver board with HDMI Shield. Connect MicroUSB port of STM32F103 to your PC.



7. With the correct driver installed you will see all buttons enabled. Make sure the **Status Window** shows 1280x720 as the LCD Resolution. If not, Click **Refresh** button.



- 8. Click **SPI Flash Tool** button at the left. You will see a dialog box for **SPI Flash Tool** with UI as below. Follow these procedures:
 - Click checkbox **128Mb (16MB)** for HDMI Shield board version 2. <u>If you have a board version 1 (with 256Mbit</u> Serial Flash), check **256Mb (32MB)**
 - Click Load *.bin File button to browse to the file 'All_76_Pic_65K.bin' we have created in step 4
 - Check **Verify** checkbox
 - Click **Auto** button



Downloading progress will start with **Erasing...** and followed by **Program** until finish.



- 9. After data download you may test it with DMA button from the menu bar with procedures as follows:
 - Click **DMA** button
 - Input 1280 as W, 720 as H.
 - Source Start Address is 1026432. This data is available from the Excel file All_76_Pic_65K.xls.
 - Source Width is 1280.
 - Click Run

RASB76/77 AP V1.0.0.0	
DMA Function	×
Destination (Memory)	□ ROM Size > 128Mb
Destination Upper-Left corner coordination Y: 0	Source (SF Fridsh)
Destination Block Width W: 1280 Destination Block Height H: 720	Source Width 1280 4
C code:	Run 5

Remarks: If you are using HDMI Shield board version 1, click ROM
Size > 128Mb because it is a 256Mb Serial Flash onboard.

Now you will see the monitor connected displaying the Pharmacy background image in no time.



Picture below shows the broccoli background with **Starting** Address changed to 2869632.



Feel free to try this software with different parameters, for example, a different **Destination Upper-Left coordinates** will display an image at another locations.

Now we have finished with the SPI Flash downloading task. Disconnect MicroUSB and HDMI cables, and disconnect STM32f103 driver board from HDMI Shield. Its task has finished.

Let's turn to the Arduino application (Ra8876_ticketdisplay_part1.ino).

Caution: To use this code on HDMI Shield Version 2 please update the Ra8876_Lite Arduino Library. Changes are:

(1) File UserConfig.h : Add support for BOARD_VERSION_2 with
128Mbit Serial Flash

(2) File Ra8876_Lite.cpp : Add support for BOARD_VERSION_2 with 128Mbit Serial Flash

We will use **Due Zipper Board** with ESP8266 Wifi SoC as the host board. Stack HDMI Shield on it with microUSB cable connected to Due Zipper, HDMI cable to a monitor.

Make sure 2.00mm jumpers all set to P12 for ESP8266.



If you don't like to use our **Due Zipper Board**, it is okay to use jumper cables to connect it to your MCU board. Example below shows an Arduino DUE board connecting with jumper cables.



Open the sketch file 'Ra8876_ticketdisplay_part1.ino'.

From [Tools] > [Board] select [Generic ESP8266 Module]. The
other parameters are illustrated below:



On **Due Zipper board**, press and hold **PROG** button, and single click on **RESET** button to bring ESP8266 to bootloader mode.

Click [Sketch] > [Upload].

Open Serial Monitor from Arduino IDE, enter any number in 3 digits you like to display, e.g. 589<Send>.





The monitor will show the queue number you just entered.

Try other numbers, say 12324545<Send>. The first 3 digits will be displayed with the remaining digits truncated.

Also try changing the source code with another background

bgIndex = putSerialFlashImage(0, 0, "background_2") >
putSerialFlashImage(0, 0, "background_1").

You will see the background changed to a with pharmacy theme.



Code description

The code starts from a description of images (**struct xImages**) that we have preloaded to Serial Flash in previous procedures. Screenshot below shows the map with reference to the All_76_Pic_65K.xls Excel file. If you change the position or content of images this map will change. You will have to change the parameters in source code to match the image parameters and addresses of the Excel file created after the change.

∞ Ra8876_scoreboard Arduino 1.8.9		All_76_Pic_65K.xls - Microsoft Excel							
File Edit Sketch Tools Help			me Insert i	Page Layout	Formulas	Data	Review	View	
		<u>∼</u> ×	Calibri - 1	1	:	F Ge	eneral 🚽		
Ra8876 scoreboard				A A I			- % ,	A	
	Pa	ste 🦪	B 2 <u>U</u> ·					Styles	
27 ///@note Source of images and their assocated binary 28 const xImages qfx assets[]=							8 ÷.0	- 6	
29 {		board 🖻	Font		Alignment		Number 🖻		
30 //digits with color background		G12 ▼ (<i>f</i> * 466560							
31 {"char0020", 108, 216, 0},		Α	В	С	D	EF	G	-	
32 {"char0030", 108, 216, 46656}, 33 {"char0031", 108, 216, 93312},	1	No. F	ile Name	Width	Height	D S Star	rt Address		
34 {"char0032", 108, 216, 139968},	2	1 0	har0020.bmp	108	216	1	(0	
35 {"char0033", 108, 216, 186624},	3		tching image	108	216	1	46650	5	
36 {"char0034", 108, 216, 233280},	4	wid	th, height, and	108	216	1	93312	2	
37 {"char0035", 108, 216, 279936},	5	add	ress paramete	ers 108	216	1	139968	3	
38 {"char0036", 108, 216, 326592}, 39 {"char0037", 108, 216, 373248},	6	5 0	harUU33.bmp	108	216	1	186624	1	
40 {"char0038", 108, 216, 373248},	7	6 0	har0034.bmp	108	216	1	233280	C	
41 {"char0039", 108, 216, 466560},	8	7 0	har0035.bmp	108	216	1	279930	5	
42 //digits with color mask	9	8 0	har0036.bmp	108	216	1	326592	2	
43 {"char0020_mask", 108, 216, 513216},	10	9 0	har0037.bmp	108	216	1	373248	3	
44 {"char0030_mask", 108, 216, 559872},	11	10 0	har0038.bmp	108	216	1	419904	4	
45 {"char0031_mask", 108, 216, 606528}, 46 {"char0032_mask", 108, 216, 653184},	12	11 c	har0039.bmp	108	216	1	466560	D	
47 {"char0032 mask", 108, 216, 655184},	13	12 0	har0020_mask.	108	216	1	513210	5	
48 {"char0034 mask", 108, 216, 746496},	14	13 0	har0030_mask.	108	216	1	559872	2	
49 {"char0035_mask", 108, 216, 793152},	15	14 c	har0031_mask.	108	216	1	606528	8	
50 {"char0036_mask", 108, 216, 839808},	16	15 c	har0032 mask.	108	216	1	653184	4	
51 {"char0037_mask", 108, 216, 886464},	17	16 0	har0033 mask.	108	216	1	699840	0	
52 {"char0038_mask", 108, 216, 933120}, 53 {"char0039 mask", 108, 216, 979776},	18	17 0	har 0034 mask.	108	216	1	746490	5	
54 //background images	19	18 0	har0035 mask	108	216	1	793152	2	
55 {"background_1", 1280, 720, 1026432} //pharmacy	£ 20	19 c	har0036 mask.	108	216	1	839808	8	
56 {"background_2", 1280, 720, 2869632} //broccoli	^k 21		har0037 mask.		216	4	886464	4	
57 };	22		har0038 mask.		216	1	933120		
	23		har0039 mask.		216	1	979770	5	
Done Saving.	24		ackground 1.b		720		1026432		
Set boot itash true	25		ackground 2.b		720		2869632		
	26		0		. 20				
			Sheet1 Sheet2	/Sheet3	1				
	Rea	1	STREET STREET	Uncers				100%	
					JPE	G image			
	ESP826	6 Module o	n COM18			-91			

From setup() RA8876 is initialized with 1280*720 (720p) in 16bit color mode. HDMI encoder is configured with a boosting factor from 720p to 1080p with DVI output mode.

Finally, a background image is displayed on screen with DMA function putSerialFlashImage().

```
172 void setup() {
173 Serial.begin(115200);
174 pinMode (RA8876 XNINTR, INPUT PULLUP);
    attachInterrupt (digitalPinToInterrupt (RA8876_XNINTR), isr, FALLING); //pending for Vsync hardware i:
175
176
177
     ///@note Initialize RA8876 starts from here
178
     if (!ra8876lite.begin(&CEA 1280x720p 60Hz)) //init RA8876 with a video resolution of 1280*720@60Hz
179
     {
180 #ifdef DEBUG LLD RA8876
181 printf("RA8876 or RA8877 Fail\n");
182 #endif
183 }
                         (1) Initialize RA8876 with resolution
184 else
                         1280*720 in 16-bit color
185 {
186
      ra8876lite.canvasImageBuffer(1280,720);
                                                  //Canvas set to the same size of 1280*720 in 16 bit-pe
187
      ra8876lite.displayMainWindow();
                                                  //align display window to the same canvas starting po:
       ra8876lite.canvasClear(color.White);
188
                                                  //set whole screen to white on startup
189
       ra8876lite.graphicMode(true);
      ra8876lite.displayOn(true);
190
                                                      (2) Initialize CH7035 HDMI encoder
191
                                                      _to boost 1280*720 to 1080p (DVI)
192
      HDMI_Tx.begin();
                                        //Init I2C
193
      HDMI_Tx.init(VIDEO_in_1280x720_out_DVI_1080p_60Hz); //map defined in .\HDMI\videoInOutMap.h
                                       //disable audio
194
      HDMI Tx.setI2SAudio(0, 0, 0);
195
       delay(1000);
196 }
197
198 bgIndex = putSerialFlashImage(0, 0, "background_1"); (3) Display a background image with DMA
199 }
```

Actually this is almost the end of this program. What we need to do is to put the host MCU to an infinite loop to listen for incoming messages. Once a valid ticket number of 3 digits or less is received with a '\n' new line character, it is stored in a buffer and the function putJumboScore() is called to display each digit decoded by DMA transfer from Serial Flash to RA8876. Video in RGB is automatically converted to HDMI by CH7035 with no software intervention once it is configured in initialization phase. Screen shot below shows the main loop of this program, which takes only 5 lines of code!

```
201 void loop() {
202 char char_buffer[4];
203 #ifdef ESP8266
204 yield(); //this is to avoid wdt reset in ESP8266
205 #endif
206 String inString = messageParser(); //blocking function here until \n received
207 sprintf(char_buffer, "%3d", inString.toInt()); //format to 3 digits
208 char_buffer[3]='\0'; //null terminate
209 putJumboScore(930,128, char_buffer); //print digits on screen
210 }
```

This is it! We have finished the first part of Ticket Display System with Arduino.

ⁱ Image Source : https://pixabay.com/photos/thermometer-temperature-fever-flu-833085/

ⁱⁱ Image Source : https://pixabay.com/photos/broccoli-vegetable-diet-food-fresh-1238250/

iiihttp://www.techtoys.com.hk/BoardsKits/HDMIshield/pdf/Getting_Started_Guide_D
TE20190711_update20190804.pdf