EEL4924 Senior Design

Preliminary Report

DIY GBA

CS Inc.

Submitted by
DIY GBA Abstract

The DIY GBA is a user friendly programmable handheld video game system that is designed to allow hobbyist to design and play their own hand held video games. The unit will feature a TFT (Touchscreen LCD Display) powered by an ARM Cortex M3 processor core running at 72MHz. The device has an integrated bootloader reducing the need for user purchases, such as external programmer. Battery operation allows the handheld to be played on-the-go without the need for restricting power cables. Software controlled backlight for greater gameplay in non-light friendly atmospheres, allowing users to change the brightness settings during gameplay without any bulky external switches. The games will be loaded onto the platform via drag and drop environment in the form of a 128KB hex files. Hex files are created using the open source arm-gcc-no-eabi Codesourcery compiler and custom makefile, later in the semester an IDE may be created for less advanced users.

Technical Objectives

Processing
- Prefer a 16 bit or greater MCU.
- Having greater than 30 MIPS.
- Prefer a 16 bit GPIO port width.
- On-board flash and ram is a necessity.
- Need a greater than 64k bytes of flash.
- Need greater than or equal to 5k bytes of ram.

Power
- Lithium Polymer battery power.
- USB Rechargable.
- At least 4 hours play time

Programmability
- An open source compiling tool chain.
- Bootloader on the chip that allows programming through USB.

Graphics

Hardware
- Greater then a 160 x 160 pixel density LCD.
• Color or at least 4bit gray scale.
• Integrated LCD Driver.
• Parallel communication bus.

**Software**

• Tile set and Sprite library.
• Scrollable screen.

**Audio**

• 3.3V operation.
• Two PWM channel sound.
• Internal Speaker.
• Headphone jack that will mute the speaker when active.
• Potentiometer volume control.

**Technology Selection**

<table>
<thead>
<tr>
<th>Processor</th>
<th>MHz</th>
<th>Flash/Ram</th>
<th>I/O pins</th>
<th>Architecture</th>
<th>Integrated USB</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM-Cortex-M3</td>
<td>72</td>
<td>128k/20k</td>
<td>43</td>
<td>32 bit</td>
<td>Yes</td>
<td>10.15</td>
</tr>
<tr>
<td>ATXmega-32a4</td>
<td>32</td>
<td>32k/4k</td>
<td>34</td>
<td>8 bit</td>
<td>No</td>
<td>5.88</td>
</tr>
<tr>
<td>ATmega644p</td>
<td>20</td>
<td>64k/4k</td>
<td>32</td>
<td>8 bit</td>
<td>No</td>
<td>7.39</td>
</tr>
</tbody>
</table>

*Table 1: uC selection table.*

The MCU selected for the Project was the ARM-Cortex-M3, in specific ST Microelectronics STM32F103RBT6. This chip had all of the peripherals that were needed to create the project. The full list is seen in the list below [1]:

- **MCU: STM32F103RBT6**, a 32-bit ARM Cortex M3 microprocessor
  - Clock Speed: 72 MHz
  - 128 KB Flash and 20 KB SRAM
  - 43 Digital I/O Pins *(GPIO)*
  - 15 PWM pins at 16 bit resolution *(PWM)*
15 analog input pins, 12-bit ADC resolution (ADC)
2 SPI peripherals (SPI)
2 I2C peripherals (I2C)
7 Channels of Direct Memory Access (DMA) (dma.h)
3 USART peripherals (USART)
One advanced and three general-purpose timers (Timers)
Dedicated USB port for programming and communications (USB)
JTAG (JTAG)
Nested Vectored Interrupt Controller (NVIC) (including external interrupt on GPIOs)
Support for low power, sleep, and standby modes (<500 μA)
Operating Voltage: 3.3 V

Especially it has 16 bit data ports that allow very quick writes to the 16bit TFT screens data port. The ARM-Cortex-M3 also has an open source tools chain that has been well documented by the open source community, the arm-gcc-no-eabi packaged nicely by Codesourcery. The reason that the STM chip was chosen was due to the open source maple development libraries that will allow for fast software development.

**Screen**
The choice of screen is determined by a list of requirements that must be met:

- 3.3V logic and backlight is preferred to minimize power supply complications.
- Must be greater than 160x160 pixels to be at least the same or better resolution than the original Game Boy.
- Prefer color, but if color is not an option at least four bit gray scale shading.

This resulted in choosing a 2.8” 320x240 pixel 18bit color TFT screen. The screen has an integrated ILI9325 driver chip, and backlight. This screen is particularly good because it is produced in large quantities out of China, allowing it to be very low cost. Also, the screen is very well supported by the open source community, this means that the graphics and interface surface does not need to be written from scratch. This should save a lot of development time.

**Charging/ Power Path**
The classic design for battery management on a handheld device is for the battery and load to be in parallel, and the charger to connect to them. This causes the issue of constant current charging being very complex if your load is constantly changing.

To solve this problem a new technology was developed called “Power Path”. This technology works by having an internal switch connect the battery and load together. That way when the charging current is applied the switch changes and
both the load and battery are connected separately. This allows you to use the device while charging, just like your cell phone.

The options are seen in the table below:

<table>
<thead>
<tr>
<th>Chip Name</th>
<th>Max Current</th>
<th>Timer</th>
<th>Battery Disconnect</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCP73871</td>
<td>1.8A</td>
<td>Yes</td>
<td>No</td>
<td>$1.94</td>
</tr>
<tr>
<td>LTC4055</td>
<td>1.6A</td>
<td>No</td>
<td>No</td>
<td>$4.38</td>
</tr>
<tr>
<td>ISL9301</td>
<td>800mA</td>
<td>Yes</td>
<td>Yes</td>
<td>$4.00</td>
</tr>
</tbody>
</table>

*Table 2: Charging IC selection table.*

This decision was completely decided by cost. The MCP73871 was significantly lower cost then any other chip. It also met all of the requirements that were specified. It would have been nice to have the built in battery disconnect, but not for an extra $2.00.

Audio

The two chips that meet the specifications are seen in the table below:

<table>
<thead>
<tr>
<th>Chip Name</th>
<th>Working Voltage</th>
<th>Watts</th>
<th>Volume Control</th>
<th>Headphone Sense</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM4865</td>
<td>2.7-5.5V</td>
<td>750mW</td>
<td>Yes</td>
<td>Yes</td>
<td>$2.47</td>
</tr>
<tr>
<td>TPA0211</td>
<td>2.5-5.5V</td>
<td>660mW</td>
<td>No</td>
<td>Yes</td>
<td>$2.45</td>
</tr>
</tbody>
</table>

*Table 3: Audio amplifier selection table.*

Both the chips provide a headphone sense pin. This pin turns off the speaker allowing only the headphone to work when the headphone jack is plugged in.

The LM4865 is selected because of the DC volume control will simplify circuit design, also it requires less external components then the TPA0211.

Product Design

The general system block diagram can be seen in the image below.
At the center of the design is the microprocessor. It's inputs are the buttons, touchscreen, usb communication, and 3.3V power.

The microprocessor outputs two PWM channels to an audio amplifier. The audio amplifier can drive both headphones and a speaker, with DC voltage controllable volume. The microprocessor also drives the LCD screen.

The power system is centered around the PowerPath chip. When charging, the chip inputs power from the USB jack, and then charges the LiPoly battery, while supply current to the load. When the USB cable is not connected the chip connects the LiPoly battery directly to the system.

**Team Work Division**
The work on the DIY GBA will be divided between individual work, and team work. A lot of the hardware work will be done by Scott, and a lot of the coding will be done by Chip.

In the table below is the list of tasks that needs to be done before the project is completed.
The list of tasks is then put into a graphical chart, as seen below in the Gantt chart.

Figure 3: Gantt Chart.

References