DESIGN PROJECT

Alarm clock

CSE 379

Dr. Kris Schindler

Spring 2015

Woo Suk Choi(woosukch)

#36517510

* Table of Contents
  + Introduction……………………………………………………... Page. 3
  + Memory Map……………………………………………………..Page.4
  + Memory Selection Table…………………………………………Page.5
  + Memory System design (Detailed hardware diagram)…………...Page.6
  + Basic Structure of AMBA interface for Alarm Clock……………Page.7
  + Decoder Diagram for ROM, RAM, Peripherals………………….Page.8
  + Design of GPIOs……………………………………………….....Page.9
    - More details about GPIO for alarm clock………………… ...Page.10
    - Port A & B for GPIO0 and GPIO1…………………………..Page.11
    - Port A for GPIO2…………………………………………….Page.12
  + Design of UART (Keypad)………………………………………Page.13
  + Components………………………………………………. …….Page.14
  + Flowchart…………………………………………………...........Page.17
  + Resources……………………………………………………… ..Page.19
* Introduction

This project is to design a digital alarm clock which is based on ARM microprocessor-based. The system has 1MB ROM and 2MB RAM. The peripherals includes 3 PrimeCell GPIOs (General Purpose Input/Output) units, 1 serial interface PrimeCell UART (Universal Asynchronous Receiver / Transmitter), and 1 PrimeCell RTC (Real Time Clock).

The clock display the time using 4 seven-segment displays (one for each digit), two LEDs to represent the colon separating the hours and minutes, another LED to indicate AM/PM, a keypad for setting the time, a buzzer (for the alarm), and four momentary push-buttons (set the time, set the alarm, turn off the alarm, and snooze).

* **Designing Memory Map for Alarm clock**

0xFFFFFFFF

Unused

0x00505000

0x00504FFF

0x00504000

RTC

0x00503FFF

0x00503000

0x00502FFF

UART

0x00501FFF

0x00502000

GPIO0

0x00500FFF

0x00501000

GPIO1

GPIO2

0x00500000

0x004FFFFF

0x003000000

0x002FFFFF

Unused

RAM

0x00100000

ROM

0x000FFFFF

0x00000000

*Figure. 1 Memory map 1MB ROM & 2MB RAM*

First, we need to design the Memory Map to figure out how much memory we need for making alarm clock.

As we see above the *Figure 1*, ROM starts from 0x00000000 to 0x000FFFFF which is 1MB, and RAM is contiguous with 2 MB which starts from 0x00100000 to 0x002FFFFF. GPIO, UART, and RTC memories are set from 0x00500000. All of peripherals have each memory of base address + offset (0xFFF) which is 12-bit each for offset. Therefore, total memory for peripherals (3 GPIO and UART) is assigned from 0x0050000 to 0x00503FFF. Also, there address line A[21] is determined only for RAM, because it never goes high until memory loads or write on the RAM location.

* **Memory Selection Table**

Memory Selection table can be distinguished between different memories. Memory selection table is shown below.

|  |  |  |
| --- | --- | --- |
| **A[21]** | **A[20]** | Memory Selection |
| 0 | 0 | ROM |
| 0 | 1 | RAM |
| 1 | 0 | RAM |
| 1 | 1 | Unused / Peripheral |

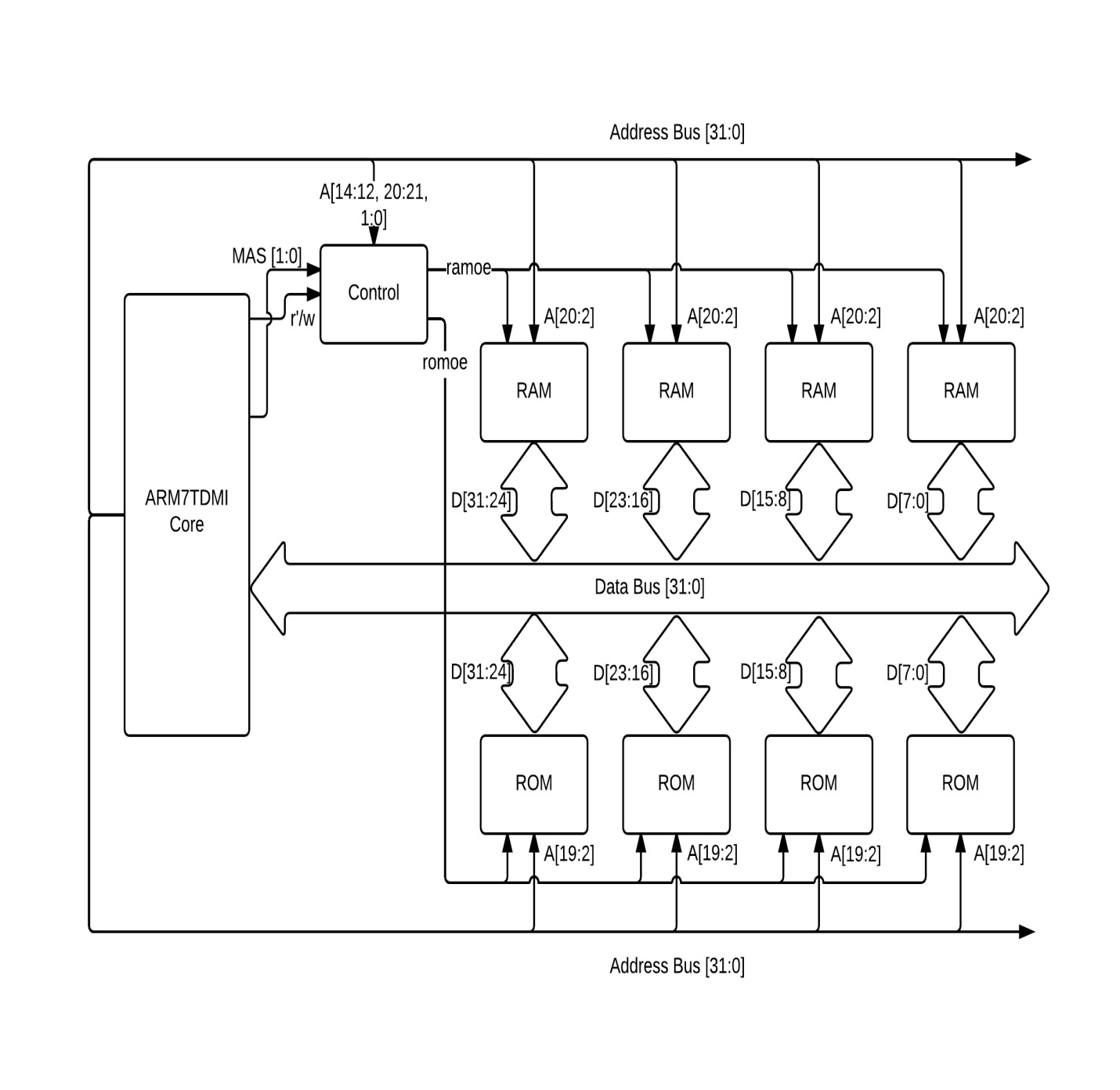
*Table 1. Memory selection to divide ROM, RAM, and Peripheral*

Address line directly linked to ROM is A[19,2]. Address line of A[20] determines that the memory would activate RAM or ROM. If it is ‘1’ assigned in address line of A[20], then it would take to RAM. It would activate ROM when you put ‘0’ for bit 20. RAM has an address line directly connected A[20, 2].

|  |  |  |  |
| --- | --- | --- | --- |
| **A[14]** | **A[13]** | **A[12]** | Memory Selection |
| 0 | 0 | 0 | GPIO2 |
| 0 | 0 | 1 | GPIO1 |
| 0 | 1 | 0 | GPIO0 |
| 0 | 1 | 1 | UART |
| 1 | 0 | 0 | RTC |

*Table 2. Other Memory Selection Address Bits for Peripherals (GPIO, UART, RTC)*

We need to separate GPIO, UART, and RTC by using other address bits. A[14], A[13], and A[12] are used to divide memory. From this information of address buses, now we can imagine the memory system design. Memory system design is shown on the next page.

* **Memory System Design (Detailed hardware diagram)**

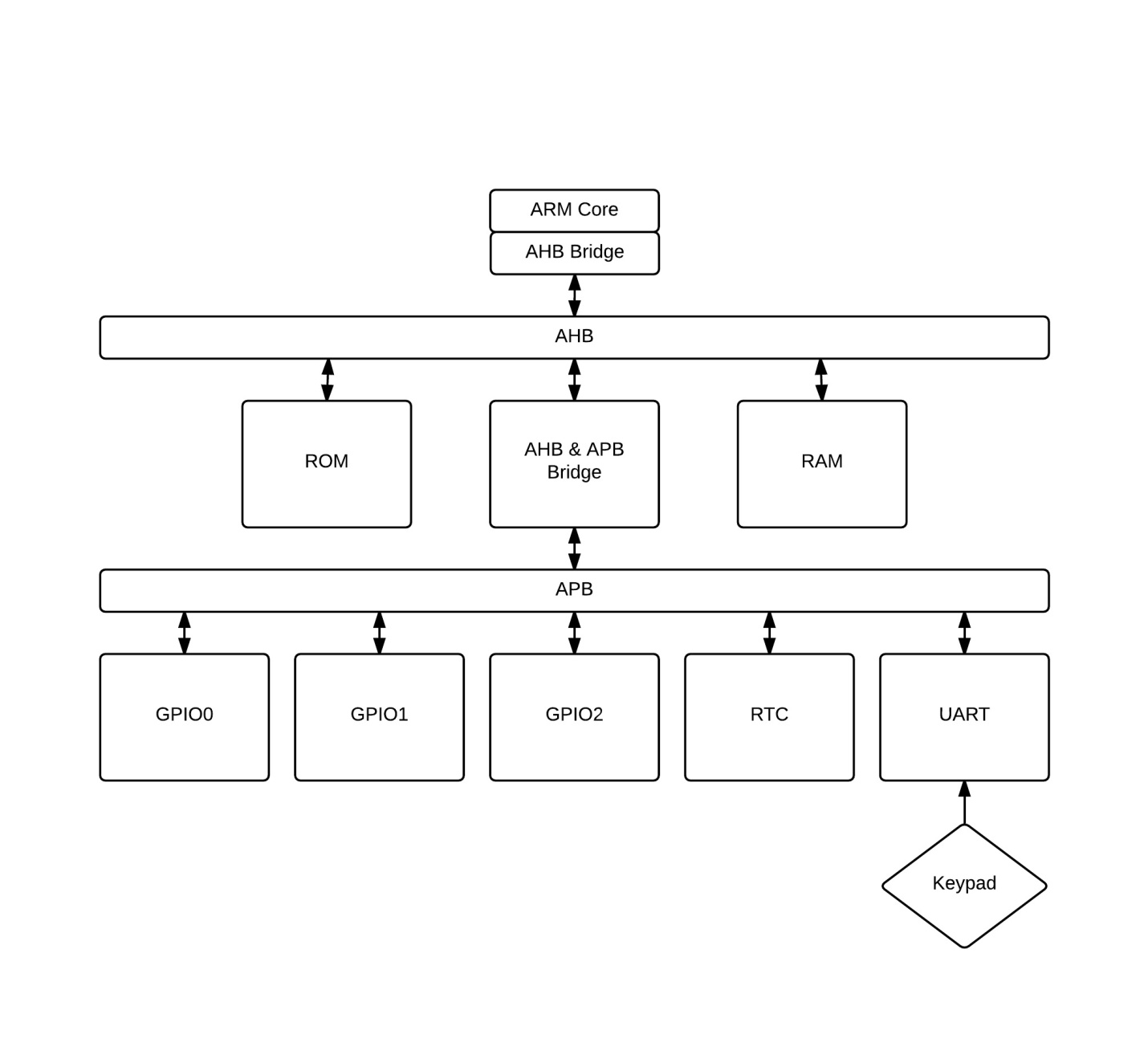
*Figure. 2 Memory System Design for 1MB ROM & 2MB RAM*

ROM is 2MB which uses the address line A[19:0] which is (bit 0 included)

RAM is 4MB which uses the address line A[20:0] which is (bit 0 included)

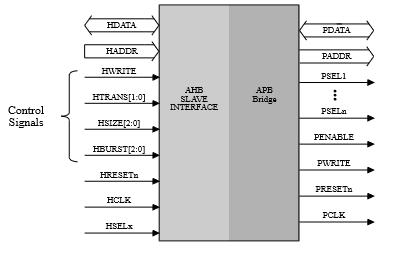
To use appropriate memory space, we need the assigned address line which can determine the operation of memory selection from control block. Address line A[14:12] and A[ are the criteria for memory selection through GPIO, UART, and RTC in above diagram.

* **Basic structure of AMBA interface for Alarm clock**



*Figure. 3 Basic diagram of AMBA Interface for alarm clock*

AHB is used to interface the processor with high performance, high clock frequency system modules such as memory, and APB is used to interface the processor with lower peripherals such as GPIOs, UART, and RTC. As shown above in the diagram, AHB is bidirectional which can store or load the data. Also, it is linked to AHB & APB Bridge to decode peripherals such as GPIO, UART, and RTC.

← *Figure. 4 APB Bridge from Kris Schindler, Introduction to Microprocessor Based System Using the ARM Processor, Second Edition, Pearson, 2013 page.159*

The decoding required for the memory-mapped I/O is contained in APB Bridge.

* **Decoder Diagram for ROM, RAM, and Peripherals**

As we see from memory map and selection table section, address lines of A[21], A[20], A[14], A[13], and A[12] are able to control and to direct to specific memory location.

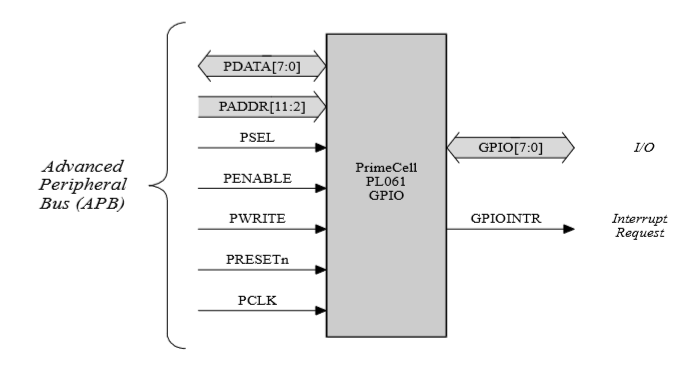
Based on the memory selection table, partial decoder diagram is in below.

*Figure 5. Decoder for RAM & ROM*

*Figure 6. Decoder for Peripherals which are GPIOs, UART, and RTC*

* **Designs of GPIOs**

The PrimeCell GPIO (PL 061) provides eight programmable input or output ports for integration into an ARM system. The GPIO module is connected to the Advanced Peripheral Bus (APB). GPIO is implemented as a memory-mapped I/O device and is selected by the APB bridges.



*Figure. 7 the interface provided by a single PL 061 GPIO between the APB and GPIO from Kris Schindler, Introduction to Microprocessor Based System Using the ARM Processor, Second Edition, Pearson, 2013 page.167*

* + **More Details about GPIO for alarm clock**

Initially, there are 16 pins for each one of GPIO (Port A: 8 pins, Port B: 8 pins), and there are 36 pins needed to build alarm clock for GPIO. We need to use total 48 pins from GPIOs to build alarm clock. Calculation is shown below why we need total 48 pins for GPIO.

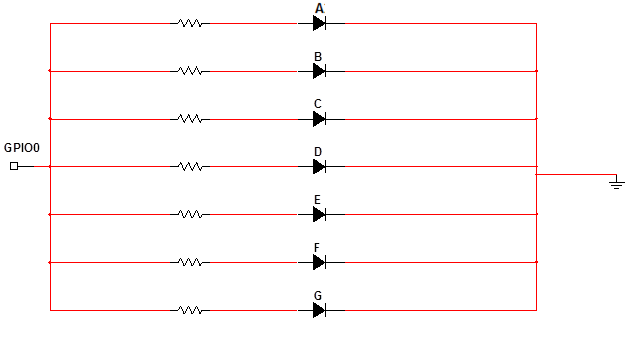
1. 7-segment display ( (4 digits for hour and minute))
2. 4-push button (4 buttons for set time, alarm, turn-off alarm, snooze)
3. LED (3 which are AM/PM LED, 2 LEDs for between hour and minute)
4. Buzzer (1)

After defining total number of pins that need for building alarm clock, assign each different purpose (Port A and Port B) into each pin to set as input or output. Table is shown below.

|  |  |  |  |
| --- | --- | --- | --- |
| **GPIO#** | **PORT** | **Purpose** | **Pin used** |
| GPIO0 | PORT A | First digit of 7-segment (hour) | 7 |
| PORT B | Second digit of 7-segment (hour) | 7 |
| GPIO1 | PORT A | Third digit of 7-segment (minute) | 7 |
| PORT B | Last digit of 7-segment (minute) | 7 |
| GPIO2 | PORT A | 3 LED (1 for AM/PM, 2 for between hour and minute) | 3 |
| 4 Push- button (set time, alarm, turn-off alarm, snooze) | 4 |
| Buzzer | 1 |
| PORT B | Unused | 0 |

*Table. 3 All of GPIO pins assigned for building alarm clock*

In GPIO0 Port A and B, we put two digits of seven-segment which setup digits of hour, so total 14 pins is used from GPIO0. In GPIO1 Port A and B, we put other two digits of seven-segment which setup for digits of minute, total 14 pins is used from GPIO1 too. For GPIO2 Port A, we use rest of LEDs (AM/PM, 2 for between hour and minute digits), 4 Push-buttons (set time, alarm, turn-off alarm, and snooze), and 1 Buzzer, so total 8 pins are used from GPIO2. Also, GPIO2 Port B is unused because Port A exactly is fit into it. After conceived all the GPIOs, let’s move onto more details of each GPIOs.

* + - ****Port A and B of GPIO0**

Segment B

Segment D

Pin 0

Pin 1

Pin 2

Pin 3

Segment C

Segment A

*Figure. 8 Circuit diagram of GPIO0*

Pin 4

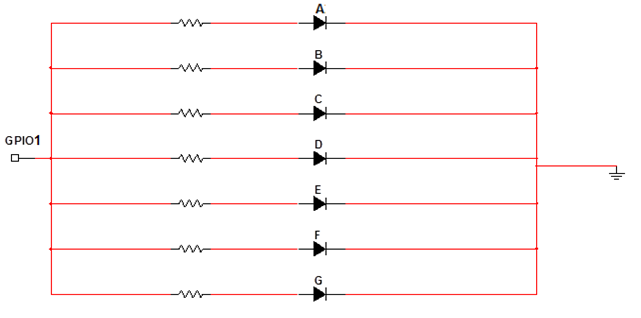
Pin 5

Pin 6

Segment G

Segment E

Segment F

GPIO 0 is connected to 7-segment display each pin indicates each part of segment such as a, b, c, d, e, f, g. Pin 0 through Pin 6 are connected to diode which indicates each segment LED. When pin is powered (high), the light is emitted. Pins which are needed to be emitted are determined in ARM assembly code.

* + - **Port A and B of GPIO1**

Pin 0

Pin 1

Pin 2

Pin 3

Segment D

Segment C

Segment B

Segment A

*Figure. 9 Circuit diagram of GPIO1*

Pin 4

Pin 5

Pin 6

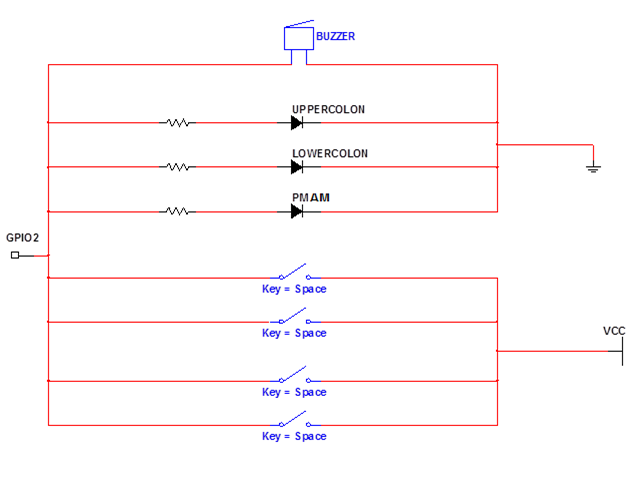
Segment G

Segment F

Segment E

GPIO1 Port A and B is same setting as GPIO0 Port A and B, because it is same seven-segment. GPIO1 is also connected to 7-segment display each pin indicates each part of segment such as a, b, c, d, e, f, g. Pin 0 through Pin 6 are connected to diode which indicates each segment LED.

* + - **GPIO 2**
      * **Port A**



Pin6

Pin7

Pin5

Pin4

Pin3

Pin2

Pin1

Pin0

AM / PM

Alarm off

Snooze

Set Time

Set Alarm

Buzzer

Upper Colon

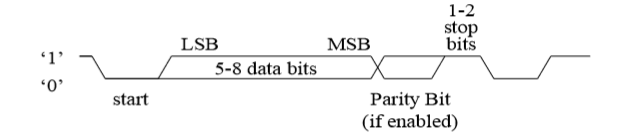
Lower Colon

*Figure 10. Circuit Diagram of GPIO2*

In GPIO 2, there are four push buttons for switches, 3 LEDs for diodes, and one for buzzer.

As we see above *Figure 10*, 4 push-buttons have functions that are setting time, alarm, snooze, and alarm off button. 3 LEDs are for illuminating lights between hour and minute digits which are same format as 7-segment display in Figure 8 in GPIO0 and GPIO1.Also, 3 LEDs and buzzer are on when pin 0 through pin 3 are high because it is grounded and current is flowing through the LED. Buzzer has a function that make a sound when time is the same as alarm time.

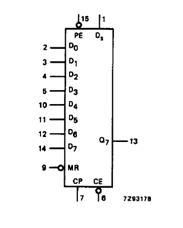
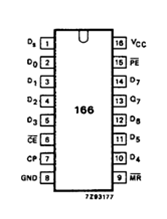
* **Design of UART**

UART is a communication device that supports serial data transfer (one bit at a time). The PrimeCell UART (PL011) provides a programmable serial interface allowing the number of data bit, parity, number of stop bits, and baud rate to be controlled by software, as shown below *Figure 11*.

*Figure 11. UART character Frame, page 2-15, PrimeCell UART (PL011) Technical Reference Manual, ARM, Revision r1p5, Document number ARM DDI 0183G, December 18, 2007*

The PrimeCell UART module connects to the Advanced Peripheral Bus (APB). It is implemented as memory-mapped I/O device and is selected by the APB Bridges

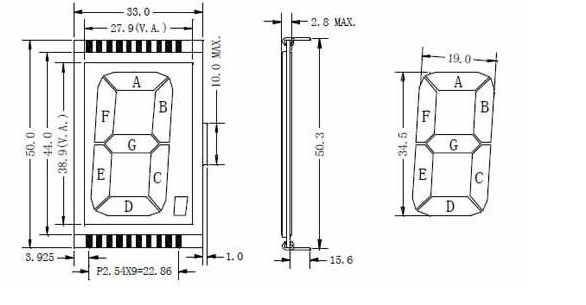
* + **Keypad**

Keypad is needed for setting the time. So, keypad is connected to UART to transmit data when the user presses keypads, but the problem in here is that the data is sent to UART parallel not in serial. So, we need a converter chip to make it sent the data properly.

*Figure. 12 Keypad and a chip for converting parallel to serial*

That chip is used to convert UART data parallel to serial which makes two high pins (which indicates row and column each) into series to make coding enable.

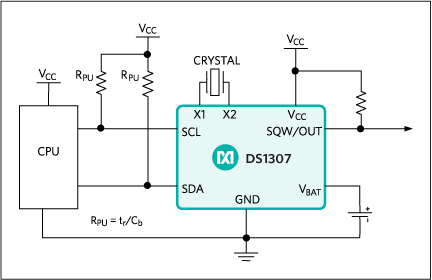
* **Components**
  + **7-segment (Part number : EDC004, Manufacturer: Aliexpress)**
    - http://www.aliexpress.com/store/product/Standard-7-Segment-1digit-LCD-Panel/600281\_374880604.html
    - Description: Seven segment (segment A through G).



*Figure. 13 picture of seven-segment (EDC004) from aliexpress*

* **LED (Part number: SSL-LX5093ID, Manufacturer: Lumex)**
  + http://www.alliedelec.com/lumex-ssl-lx5093id/70127533/
  + Description: Red LEDs for illuminating

*Figure. 14 picture of Red LED (SSL-LX5093ID) from Lumex*

* + **RTC (Part number: DS1307, Manufacturer: Maxim Integrated)**
    - [**http://www.maximintegrated.com/en/products/digital/real-time-clocks/DS1307.html**](http://www.maximintegrated.com/en/products/digital/real-time-clocks/DS1307.html)
    - Description: The DS1307 serial RTC is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM.

*Figure. 15 picture of RTC from maxim integrated*

* + **Buzzer (Part number: K16-CZL, Manufacturer: Xinmaosheng)**
    - [**http://www.alibaba.com/product-detail/New-crazy-selling-alarm-clock-buzzer\_1788775728.html**](http://www.alibaba.com/product-detail/New-crazy-selling-alarm-clock-buzzer_1788775728.html)
    - Description: Buzzer for alarm clock.

*Figure. 16 picture of Buzzer from alibaba*

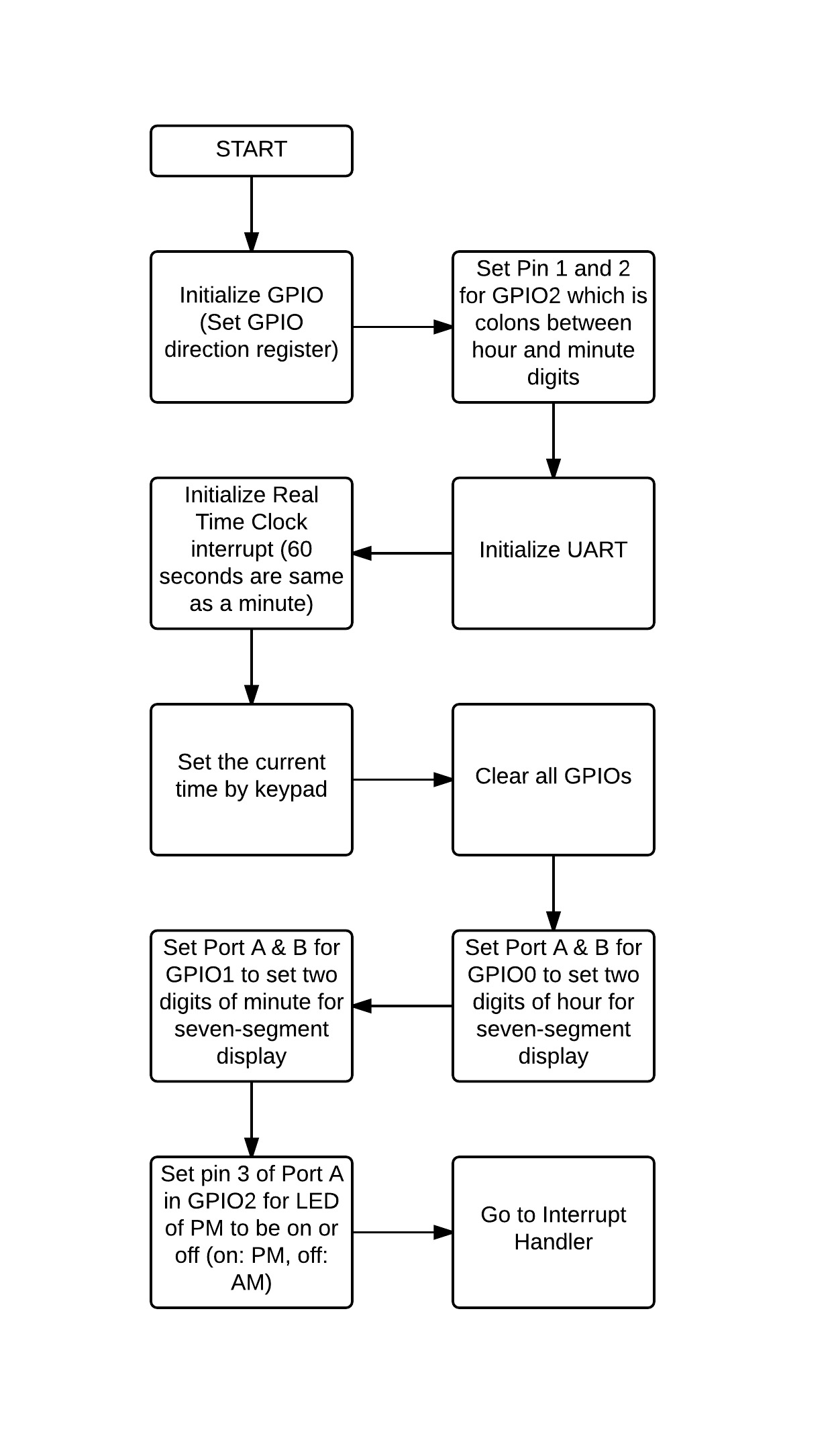
* + **Push button (Part number: 8500, Manufacturer: ITT industry)**
    - ****[**http://www.alldatasheet.com/datasheet-pdf/pdf/161499/ITT/8500.html**](http://www.alldatasheet.com/datasheet-pdf/pdf/161499/ITT/8500.html)
    - Description: push-buttons for setting time, alarm, snooze, alarm-off.

*Figure. 17 picture of push-buttons from alldatasheet.com*

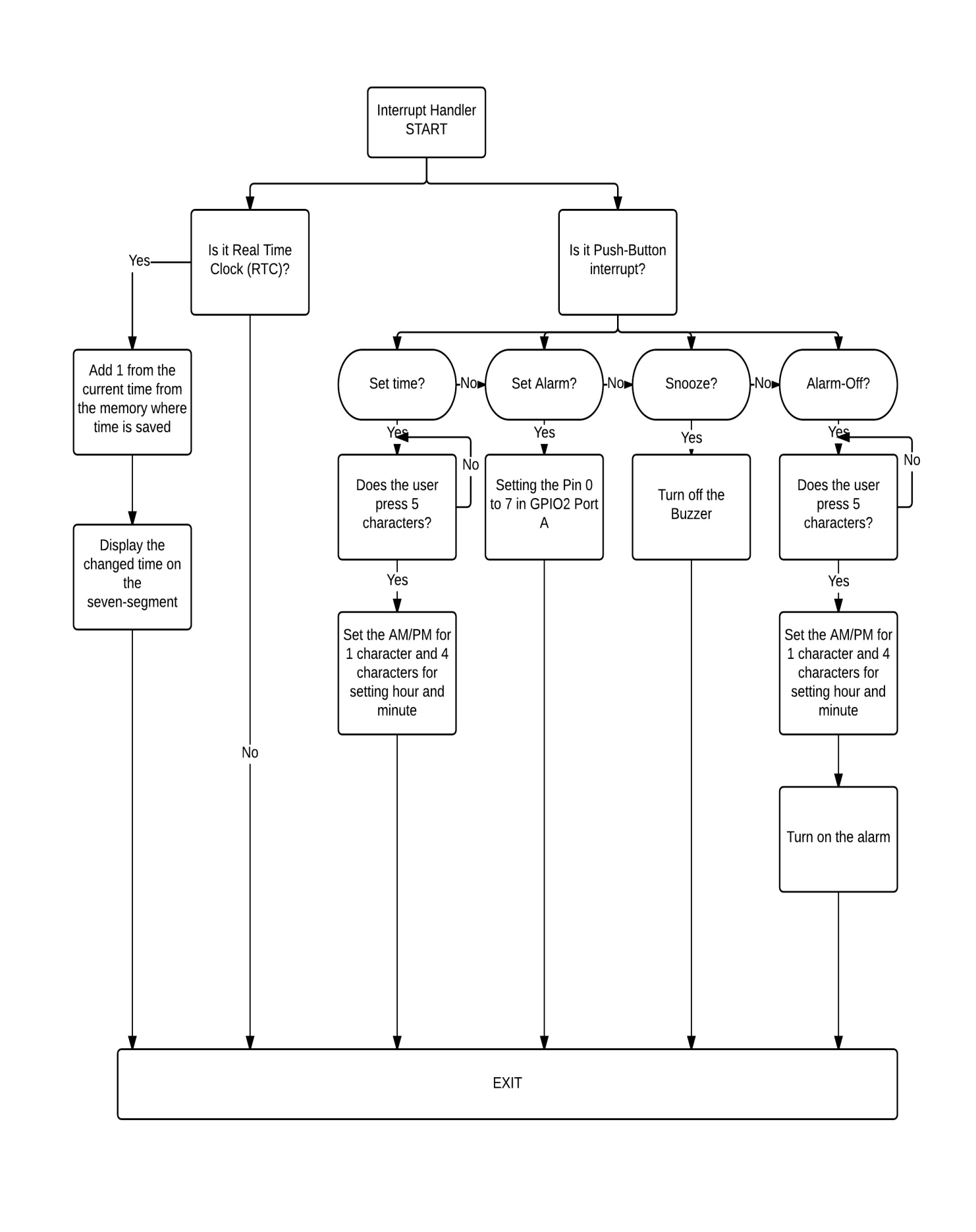
* + **Keypad (Part number: 27899, Manufacturer: Parallax)**
    - [**https://www.parallax.com/product/27899**](https://www.parallax.com/product/27899)
    - [](https://www.parallax.com/sites/default/files/styles/full-size-product/public/27899.png?itok=zmmewvUT)Description: Keypad for setting the time. 8pin access to 4x4 matrix

*Figure. 18 picture of Keypad from parallax.com*

* **Flow Chart**
  + **Initialization**



* + **Interrupt Handler**

****

* **Resource**
  + Introduction to Microprocessor Based System using the ARM Processor, Second Edition, Pearson, by Dr. Kris Schindler.
    - Page 159, APB Bridge (Figure. 4)
    - Page 167, the interface provided by a single PL 061 GPIO between the APB and GPIO (Figure. 7)
  + ARM Technical Reference
    - ARM PrimeCell UART(PL010) Technical Reference Manual
    - ARM PrimeCell GPIO(PL060) Technical Reference Manual
    - ARM PrimeCell RTC(PL031) Technical Reference Manual