**BUILDING A DIGITAL CLOCK USING LPC2148 MICROCONTROLLER**

A PROJECT REPORT

*Submitted by*

**LAVANYA SIRISHA BL.EN.U4CSE19076**

**TRISHA CHANDER BL.EN.U4CSE19133**

**VINAMRATHA PATTAR BL.EN.U4CSE19145**

***for the course***

***19CSE303- Embedded Systems***

***Evaluated by***

***Ms. Radha D***



**AMRITA SCHOOL OF ENGINEERING, BANGALORE**

**AMRITA VISHWA VIDHYAPEETHAM**

**BANGALORE-560 035**

November-2021

**BUILDING A DIGITAL CLOCK USING LPC2148 MICROCONTROLLER**

**Project Description :**

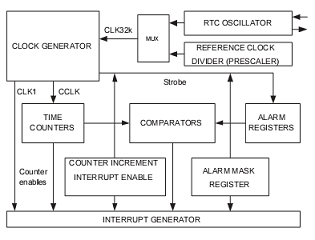
LPC2148 is the widely used IC from the ARM-7 family. It is a reliable option for beginners as well as high-end application developers. Real Time Clock (RTC) is used to store Time and Date in the system even when the system is not in operation. This is the system used in many devices including Laptops, Mobile phones, tablets, Digital Cameras, etc. RTC is an inbuilt part of any electronic device. DS1307 is a widely used RTC which can interface with any controller. LPC2148 controller now has an inbuilt RTC to store timing information. This is a battery backup power clock so that it tracks the time even while the computer is turned off, or in a low power state. It is an IC (not a physical clock) that is present on the motherboard and responsible for timing the functioning of the system and system clock.

Our project’s aim is to configure this RTC and display time information onto the LCD which can be integrated into multiple devices. Any computer device has two clocks - a hardware clock known as the Real Time Clock and the other is Software Clock. We configure it such that the lcd accepts data (from D0-D7 datapins) or commands from the KEIL program and displays the time (in terms of Seconds, Minutes, Hours) which can further be developed to use as a component responsible for interrupts, timer, task scheduling and synchronization, etc.

**Software and Hardware tools used:**

* *KEIL MicroVision 4:* This is a software development platform that can be used to create and build embedded applications such as a C/C++ compiler, macro assembler, etc. It is equipped with a device database, project manager, dialogs and even a flash programming utility. The platform offers a build mode to create applications and a debug mode for debugging the applications. With respect to this project, we have coded in KEIL’s C programming IDE using LPC2148 as our microcontroller.
* *Proteus Design Suite:* This software is used to simulate, design and build drawing circuits. It’s primary use is for electronic circuit simulation. It is also used as the design phase of a PCB layout project. It’s a core component and is included with all product configurations. The microcontroller simulation in Proteus works by applying either a hex file or a debug file to the microcontroller part on the schematic.
* *LPC2148:*  One of the most widely used IC from the ARM-7 family, LPC2148 is preloaded with several inbuilt peripherals. Its RISC based approach leads to reduced costs, heat, power use along with several other advantages like portability. As mentioned earlier, it’s processor is ARM7 – TDM-S and supports instructions that are 32-bit ARM or 16-t Thumbbi. It has upto 45 GPIO pins. LPC2148 has two ports for input output operations – PORT0 and PORT1. PORT0 contains pins for individual direction and PORT1 has pins for bidirectional purpose. LPC2148RTC can be clocked by a separate 32.768 KHz oscillator or by a programmable prescale divider based on the APB clock. It maintains a calendar and clock and provides seconds, minutes, hours, month, year, day of week, day of month and day of year.It has power supply pin that can be connected to a battery or to the main 3.3V, during power down mode, it uses little power.
* *LM016L:* This is a 16x2 LCD component used in our project. This is the most commonly used LCD based on Hitachi’s HD44780 controller.16 x 2 basically means that it can display 16 characters per line and that there are 2 lines. LM016L has 16 pins. Pin 1 is for power supply(GND) and pin 2 is for power supply(+5V). Pin 4 is the RS pin (register select) which is basically a control pin to specify if the instruction register or the data register should be connected to the data bus. Pin 5 is 0 to indicate writing operation to the LCD Module and 1 if the LCD Module has to be read. Pin 6 denotes the enable which is used to latch information presented to its data pins. Pin 8 to pin 14 are the data bus lines.

**System model**



**Figure 1: Block Diagram for RTC Configuration**

So, RTC keeps track of time by counting the cycles of the 32kHz crystal oscillator circuit, an internal capacitor-based oscillator, or even an embedded quartz crystal. The RTC has two types of clock sources - the external oscillator and an internal pre-scalar clock as mentioned. To select the internal Prescaler, the PREINT and PREFRAC register values must be calculated and must be updated. After this, RTC must be reset by setting the CLCRST bit in the CCR register.

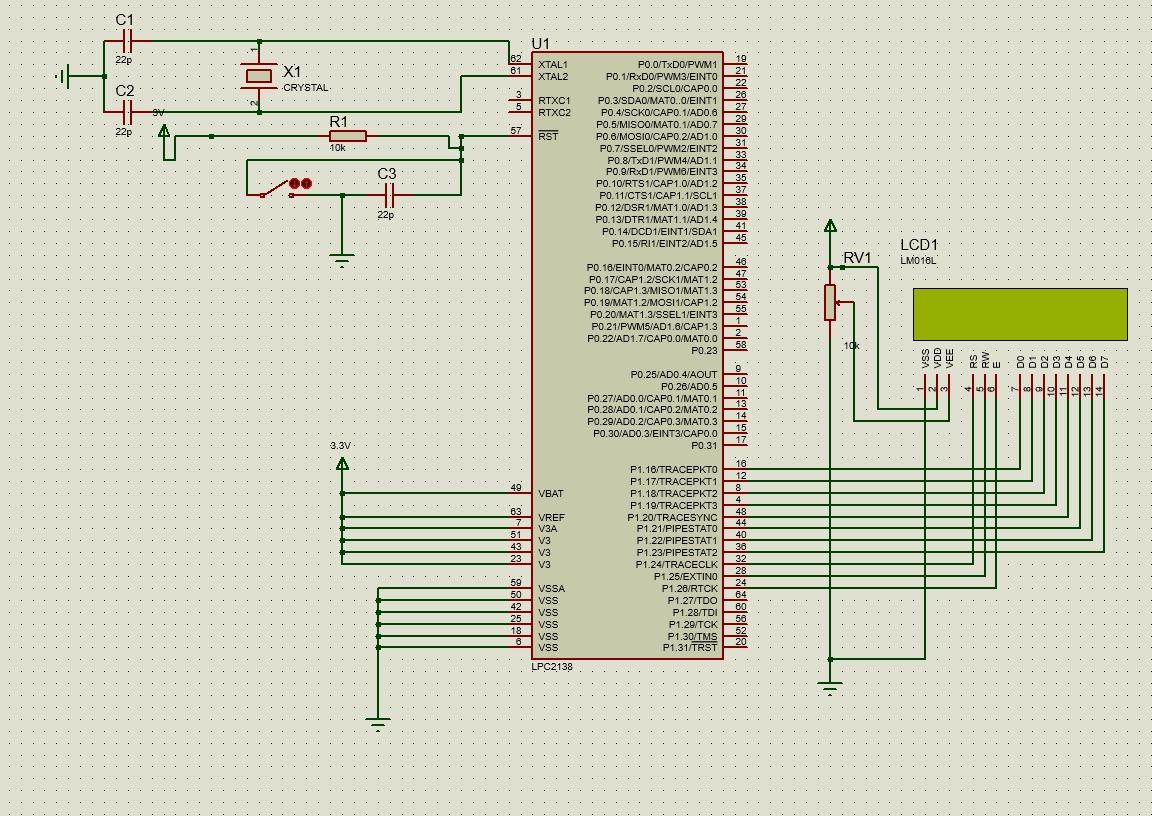
PREINT = int (PCLK / 32000) ? 1.

PREFRAC = PCLK ? ((PREINT + 1) × 32000).

There are four main groups of registers for RTC :-

* Miscellaneous registers - Interrupt Location Register(ILR), Clock Tick Counter(CTC), Alarm Mask Register(AMR);
* Time Counter register Group - registers that give Time values [Hour, Minute, Second].
* Alarm register group – registers to hold alarm settings. When a match occurs, an interrupt is generated;
* Registers that control Reference Clock Divider – we get 32768Hz from peripheral clock frequency using these registers.

An interrupt must occur every time any timer-counter is incremented, so an interrupt every second and once a minute/hour when the minute/hour value is incremented. So this way, you can generate interrupts for each of the eight-time registers [of CTIME0, CTIME1, CTIME2] increment.



**Figure 2: Circuit in Proteus**

The prescaler basically allows the timer to be clocked at the rate that we want. It essentially reduces a high frequency electrical signal to a lower frequency by integer division. We brought it down from 32MHz to 12Mhz.

**Implementation Details**

#include<lpc21xx.h>

#define rs (1<<24)

#define rw (1<<25)

#define en (1<<26)

void delay(int j )

{

int i;

for(;j;j--)

for(i=6000;i;i--);

}

void data\_lcd(char ch)

{

int i =0;

i = ch;

i = i<<16;

IOPIN1 &=(0XFF00FFFF);

IOPIN1 |= i;

IOSET1 = rs;

IOCLR1 = rw;

IOSET1 = en;

delay(2);

IOCLR1 = en;

}

void cmd\_lcd(char ch)

{

int i =0;

i = ch;

i = i<<16;

IOPIN1 &=(0XFF00FFFF);

IOPIN1 |= i;

IOCLR1 = rs;

IOCLR1 = rw;

IOSET1 = en;

delay(2);

IOCLR1 = en;

}

void init\_lcd()

{

cmd\_lcd(0x38);

cmd\_lcd(0x01);

cmd\_lcd(0x06);

cmd\_lcd(0x0c);

cmd\_lcd(0x80);

}

void str\_lcd(char \*str)

{

while(\*str)

data\_lcd(\*str++);

}

void time(void)

{

cmd\_lcd(0x80);

str\_lcd("HH:MM:SS");

cmd\_lcd(0xc0);

data\_lcd(48+(HOUR/10));

data\_lcd(48+(HOUR%10));

data\_lcd(':');

data\_lcd(48+(MIN/10));

data\_lcd(48+(MIN%10));

data\_lcd(':');

data\_lcd(48+(SEC/10));

data\_lcd(48+(SEC%10));

}

void SetTime(void)

{

CCR = 0x02;

HOUR = 10;

MIN = 50;

SEC = 0;

CCR = 0x11;

}

int main(void)

{

SetTime();

PINSEL2 = 0X00000000;

IODIR1 = 0XFFFFFFFF;

init\_lcd();

while (1)

{

time();

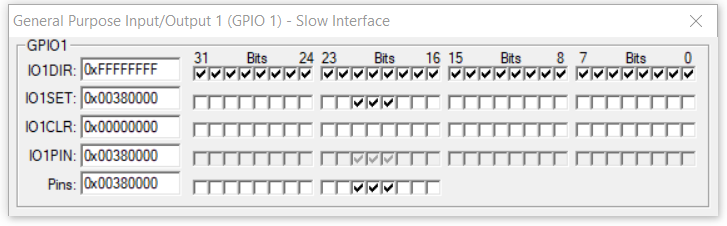
}

}

**Sample Output:**

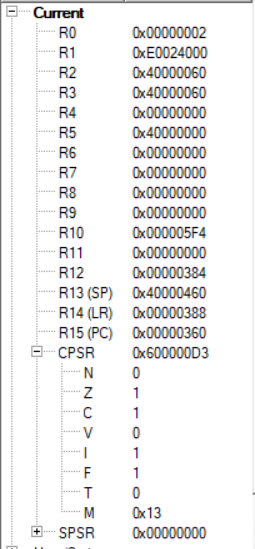
KEIL peripheral and registers output and proteus output (snap shots)

**KEIL PERIPHERAL OUTPUT -**

****

**Figure 3: Command code 0x38 being set (to set up the LCD display)**

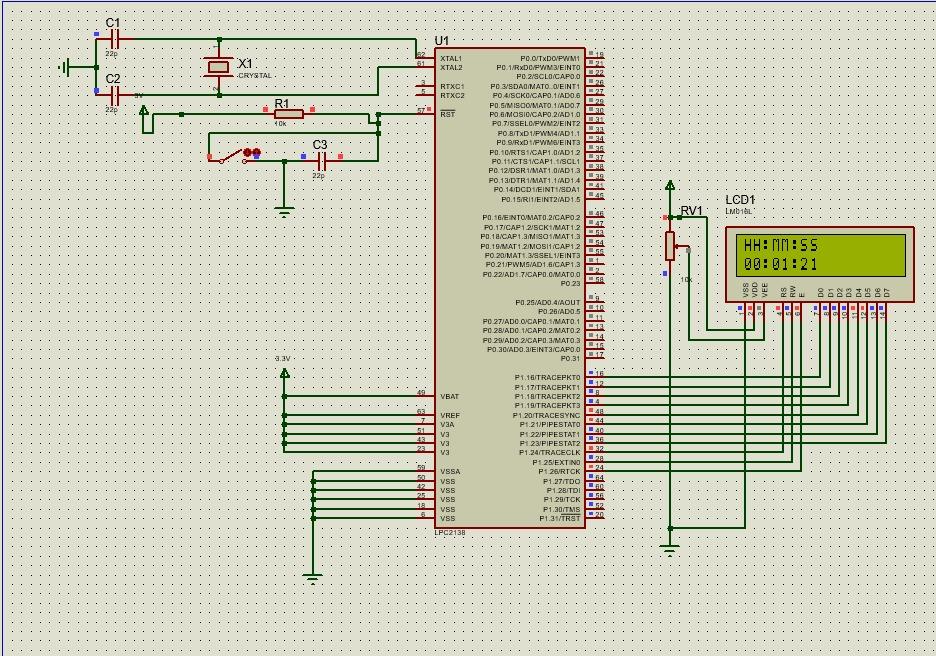
**KEIL REGISTER OUTPUT -**

****

**Figure 4: Bit Address of Time Group Register in LPC2148 - R1**

CCR=0X2 is stored in R0. Similarly the addresses for the time group of registers are stored from 0xE0024000 (Hour) to 0xE0024008(seconds). The values of hour, minute, second change in the R0 register itself.

**PROTEUS OUTPUT**

****

**Figure 5: RTC Output in Proteus**

As shown, our proteus model displays the current time as a digital clock using RTC which has additional power supply pins (3.3V) that allow the RTC clock to work when the system is off – actually power-down mode operated from the clock battery. On pressing the switch, you can halt the timer for the specified time. Additionally, for future development, we can add the date registers to make it work like a calendar as well.