HOME AUTOMATION USING BLUETOOTH

TEAM MEMBERS

SHREYASH ARYA(2015097)
TUSHITA RATHORE(2015108)
AKHIL GOEL(2015126)
SARTHIKA DHAWAN(2015170)

PROJECT SCOPE

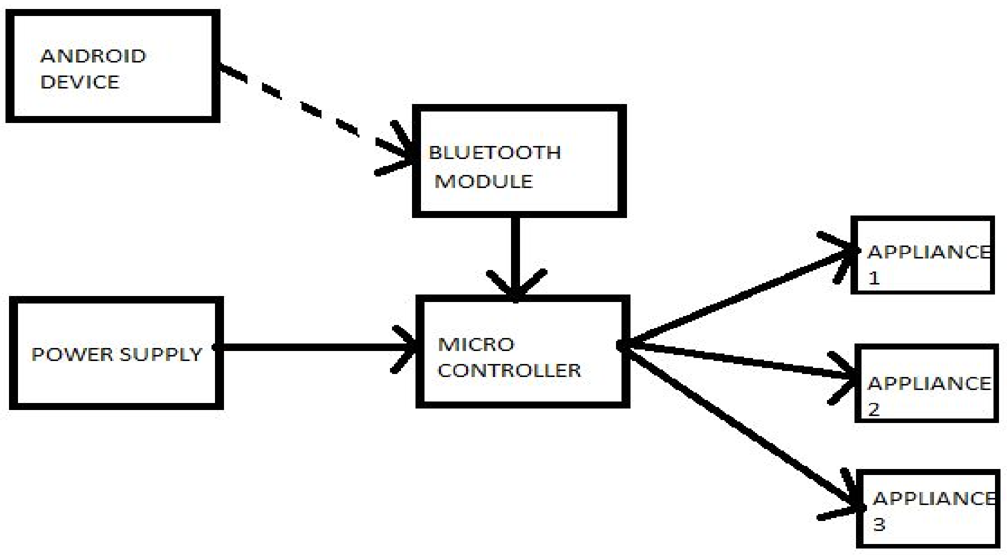
The project lives upto the reputation of technology. We have build a home automated system which uses and incorporates Computer Organisation concepts.

Usually conventional wall switches are located in different places in the house and require someone to help turn the loads on and off. For the elderly and physically handicapped people, the task of switching on and of the switches is quite difficult. Therefore, the system is used to control the home appliances using an android app.

COMPONENTS -

1. ATMEGA328P Microcontroller (with resistors , capacitors etc)
2. Bluetooth module (HC-05)
3. LEDs
4. Android phone with bluetooth controller app
5. Wires

DESIGN -



Description:

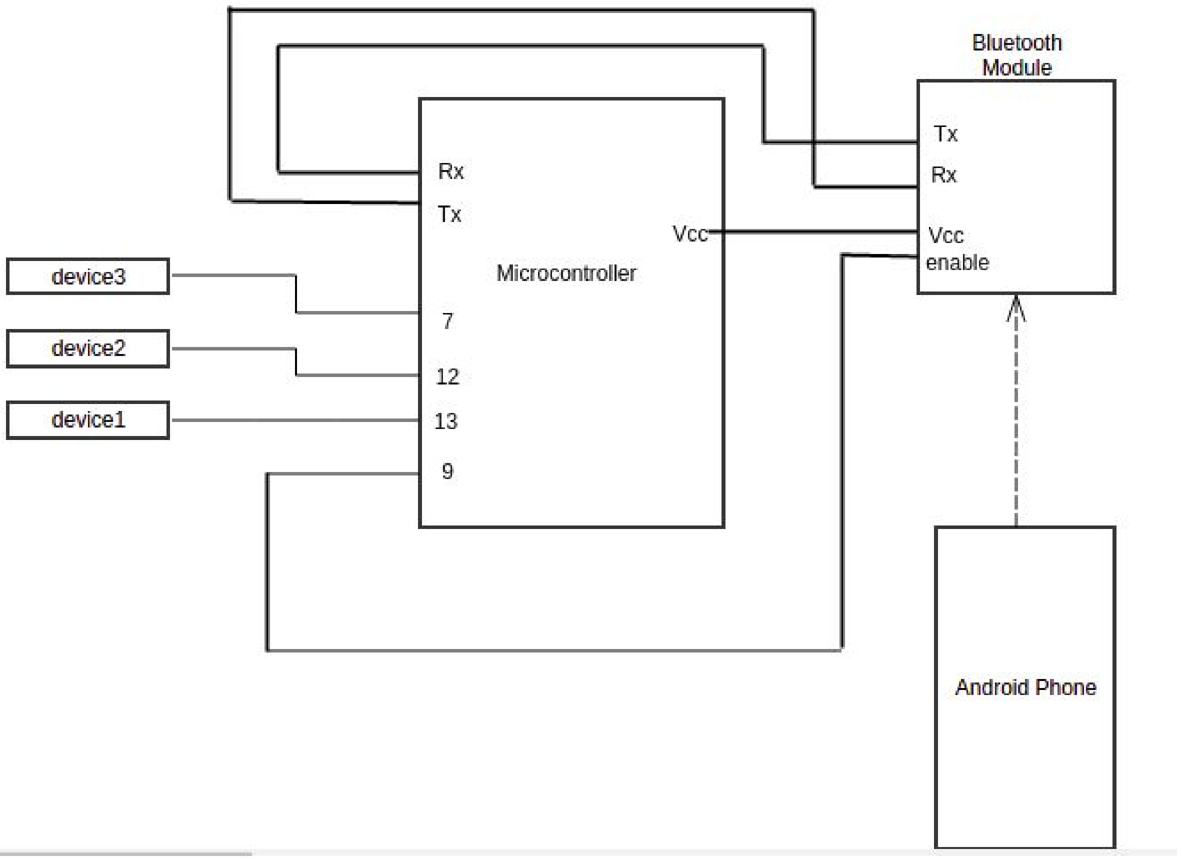
User with the android phone decides which device he wants to operate and sends the device number accordingly via bluetooth controller app. The bluetooth module(HC-05) receives the data and sends it to the microcontroller using serial ports. The data sent is then processed and the microcontroller(based on the received data) issues an interrupt to the appropriate device.

Entering 1 on the android app - Switches on the LED on Microcontroller (pin 13 - appliance 1)
Entering 1 again - switches off this LED

Entering 2 on the android app - Switches another LED (pin 12 - appliance 2)
Entering 2 again - switches off this LED

Entering 3 on the android app - Switches on the last LED (pin 7 - appliance 3)
Entering 3 again - switches off this LED

INTERFACE DIAGRAM -



MILESTONES -

1. Understanding ATMEGA328P architecture.
2. Learning embedded C.
3. Getting familiar with HC-05 bluetooth module.
4. Writing a working piece of code and debugging it.
5. Improving the code using various Computer Organisation concepts like interrupts, register addressing, timers, power optimization, byte and bit addressing, memory management etc.

COMPUTER ORGANISATION CONCEPTS USED -

CO Concept -	**Usage in the code -**	**Used for -**
Timers	LINE 1	Providing delay
Byte Addressing	LINE 2	Setting the value of a complete 8 bit register at once
Immediate register addressing	LINE 3	Storing values in registers. The operand on LHS may be a register or memory location, and the operand on RHS is an immediate constant
Power saving	LINE 4	Switching off unnecessary functionalities that our program does not use
Bit addressing	LINE 5	Setting/Unsetting a single bit of a register
Memory Allocation in RAM	LINE 6	The variables that we use are temporarily given memory in RAM.
Interrupt	LINE 7	Whenever HC-05 receives data, an interrupt is triggered
Memory Allocation in Flash	Whole code (1148 bytes)	The code that we burn is stored in the flash memory
Serial Ports	Architectural Concept	Our module communicates with the microcontroller via the serial ports
Reset	Architectural Concept	In case the system hangs, the reset button puts the **program counter** to the start location of the program.

CODE-

```
#define F_CPU 16000000UL

#include <avr/io.h>
#include<avr/interrupt.h>

char value[2];

void uart_init(uint32_t baudrate)
{
	uint16_t brate = 0;
	brate=(F_CPU/(16UL*baudrate)) - 1;
	UBRR0H = brate >> 8;                                    //UART COMMUNICATION
	UBRR0L = brate;
	UCSR0C &= ~(_BV(UMSEL01) | _BV(UMSEL00));
	UCSR0C &= ~(_BV(UPM01) | _BV(UPM00));
	UCSR0C &= ~_BV(USBS0);
	UCSR0C &= ~_BV(UCSZ02);
	UCSR0C |= _BV(UCSZ01) | _BV(UCSZ00);
	UCSR0B = _BV(RXEN0) | _BV(TXEN0);
}


uint8_t uart_getchar()
{
	loop_until_bit_is_set(UCSR0A, RXC0);
	return UDR0;
}

void uart_read_line(uint8_t len)
{
	uint8_t i;
	//PORTB|=32;
	for (i = 0; i < len; i++)
	{
		value[i] = uart_getchar();
		if(value[i] == '\r')
		{
			value[i] = '\0';
			break;
		}
	}
```

```
        if(value[0]=='1'||value[0]=='2' ||value[0]=='3')
        {
                PORTD&=~(4);
                TCNT1H=253;                                     //1. TIMER
                TCNT1L=142;                                     //2. BYTE ADDRESSING
                TCCR1B=4;
                while((TIFR1 & (TIFR1|1))==0);
                TIFR1|=1;
                TCCR1B=0;                       //3. IMMEDIATE REGISTER ADDRESSING
                //_delay_ms(10);
                PORTD|=4;
        }

}


int main(void)
{
        PRR|=225;                                               //4. POWER SAVING
        uart_init(38400);
        sei();
        DDRD|=128;
        DDRB|=16;
        DDRD|=4;
        PORTD|=4;
        EIMSK |= _BV(INT0);                                     //5. BIT ADDRESSING
        EICRA |= _BV(ISC01);
        DDRB|=32;
        PORTB|=32;
        PORTB|=16;
        PORTD|=128;
        while(1)
        {
                memset(value, 0, 2);            //6. MEMORY ALLOCATION IN RAM
                uart_read_line(1);
        }
}

ISR(INT0_vect)                                                  //7. INTERRUPT
{
        if(value[0]=='1')
        PORTB^=32;  //TOGGLE
        else if(value[0]=='2')
```

```
            PORTB^=16;
            else if(value[0]=='3')
            PORTD^=128;
}
```

RESULTS -

Hence, we are able to control the respective appliance (LED’s) by their numbers.

CONCLUSION-

Using the concepts of Computer Organization, we are able to get the appliances (LED’s for prototype purpose) controlled by the bluetooth command through android app. It can also be implemented in the real life scenario by connecting AC devices via a DAC converter.