Marble climbing machine based on camshaft principle

Student: Tiplea Teodor
Email: teodor.tiplea@gmail.com

1. Summary

The purpose of this project is to create a complex movement using one single moving part. In order to achieve this goal, knowledge about the working principle behind a camshaft must be achieved.

2. Camshaft Basics

The key part of any camshaft are the lobes. As the camshaft spins, the lobes open and close the intake and exhaust valves in time with the motion of the piston. When the intake valve opens and the piston starts its intake stroke, the air/fuel mixture in the intake runner starts to accelerate into the cylinder. By the time the piston reaches the bottom of its intake stroke, the air/fuel is moving at a pretty high speed. If we were to slam the intake valve shut, all of that air/fuel would come to a stop and not enter the cylinder. By leaving the intake valve open a little longer, the momentum of the fast-moving air/fuel continues to force air/fuel into the cylinder as the piston starts its compression stroke. So the faster the engine goes, the faster the air/fuel moves, and the longer we want the intake valve to stay open. We also want the valve to open wider at higher speeds -- this parameter, called valve lift, is governed by the cam lobe profile.

3. Solution

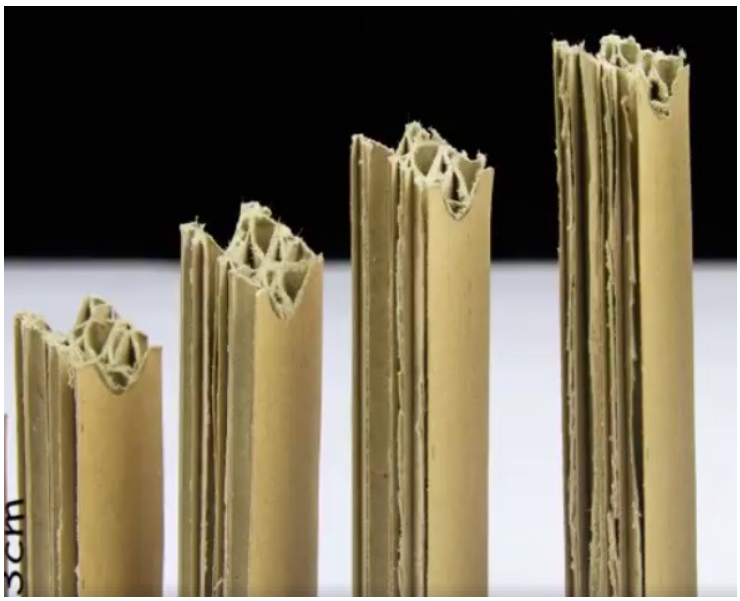
In order to describe the solution for this project, we must divide the current chapter in three smaller chapters:

A. Machine construction

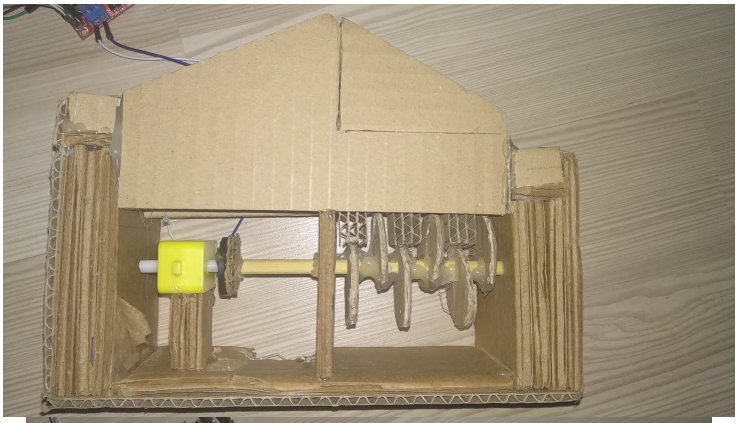
We begin this project by creating a simple version of a camshaft. Materials used for this

camshaft are cardboard discs and a wood stick. This camshaft is presented in the next picture.

The next part that is of interest is what this camshaft activates. In an engine a camshaft moves valves in a cylinder. Because this is proof-of-concept project, I have chosen to use some sort of cardboard steps. When moved up and down, the steps will move the marbles from a lower position to a higher one.

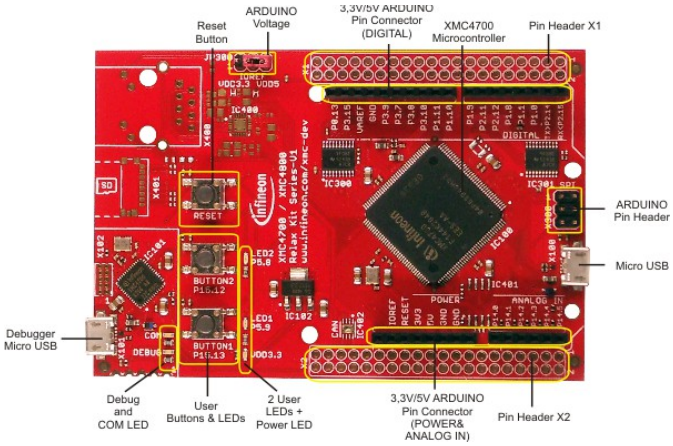


For the body of this machine I have chosen to use a structure made out of cardboard, and the result is presented in the following images:



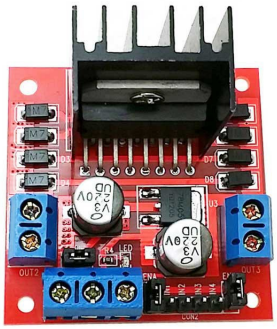
B. MCU, motors, and electrical components

- **Infineon XMC4700**

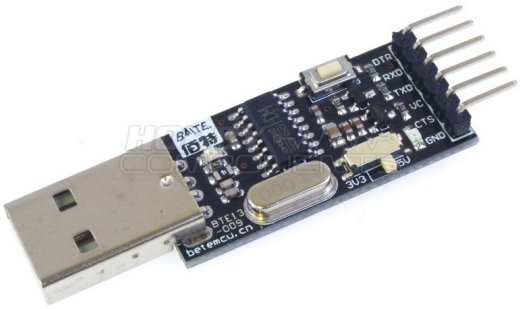


- **RaspberryPi 3 B**

- **Full-Bridge Motor Driver Dual - L298N**



- **USB to serial adapter**



- **DC motor**

The DC motor is connected to the camshaft. It is the only moving part of the entire system and it is controlled by the XMC board using the L298N dual motor driver. The direction of the motor is given by the values of In3 and In4 on the driver and the speed is given as PWM.

The L298N driver is powered by a 9V battery and the ground pin is connected to a ground pin of the XMC4700 board in order to avoid the PWM value to be transmitted with errors. The communication between RaspberryPi and XMC board is done using the USB to serial adapter. From the RaspberryPi the user can choose between 4 different speed values of the motor, one of which is the stop condition.

C. Software implementation

The XMC4700 board was programmed in the DAVE 4.3.2 IDE. The pin allocation is presented in the following image:

APP Instance Name	APP Pin Name	Pin Number (Port)
in3		
	pin	#14 (P3.7)
in4		
	pin	#13 (P3.8)
k1		
	pin	#49 (P15.13)
k2		
	pin	#50 (P15.12)
led1		
	pin	#57 (P5.9)
led2		
	pin	#58 (P5.8)
PWM_0		
	PWM Output Pin	#112 (P1.0)
UART_0		
	Receive Pin	#142 (P0.4)
	Transmit Pin	#141 (P0.5)

Explanations:

- *In3 in in4 sets the direction for the DC motor*
- *K1 and k2 are the push buttons on XMC board. Using those buttons the user can control the system in the same manner as it he can do by running the python scripts on RaspberryPi*
- *Led1 and led2 are indicators for the state of the machine*
- *PWM_0 sets the speed of the motor*
- *UART_0 is the serial communication interface with the RaspberryPi*

Next, the source code running on the xmc board will be presented:

```
#include <DAVE.h>

void delay()
```

```
{
int i;
   for (i=0;i<0x0fffff;i++); // aprox 0.5 secunda
}

int main(void)
{

   DAVE_STATUS_t status;
   uint8_t car;
   uint32_t k1_status, k2_status;
   uint32_t duty_cycle;
   status = DAVE_Init();
   int motor=0;
   int flag=0;
   UART_SetRXFIFOTriggerLimit(&UART_0, 0);

   PWM_Init (&PWM_0);

    PWM_SetFreq(&PWM_0,50);

                              // perioada 20 milisecunde
    PWM_SetDutyCycle(&PWM_0,0);          // nu genereaza semnal
    PWM_Start(&PWM_0);
    motor=1;

    // porneste pwm cu factor zero
    DIGITAL_IO_SetOutputLow(&led1);   // LED vizualizare
    DIGITAL_IO_SetOutputLow(&led2);

    // Secventa initializare ESC (Electronic Speed Controller)


    DIGITAL_IO_SetOutputLow (&in3);
    DIGITAL_IO_SetOutputHigh (&in4);
//    duty_cycle=10000;
//    PWM_SetDutyCycle(&PWM_0,duty_cycle);
    while(1)     // doar un singur LED aprins
         {
     k1_status = DIGITAL_IO_GetInput(&k1);
     k2_status = DIGITAL_IO_GetInput(&k2);
     if(k1_status == 0) // apasat
     {
            if(motor==0)
            {
                    PWM_Start(&PWM_0);
                    motor=1;
           }
          if(flag==0)
                   duty_cycle=3500;
          if(flag==1)
                   duty_cycle=4000;
```

```
        if(flag==2)
                duty_cycle=5000;
        PWM_SetDutyCycle(&PWM_0,duty_cycle);
        DIGITAL_IO_SetOutputHigh(&led1);
        delay();
        DIGITAL_IO_SetOutputLow(&led1);
        flag=(flag+1)%3;
}
        if(k2_status==0)
        {

                PWM_Stop(&PWM_0);
                motor=0;
                flag=0;

        }

        if(UART_GetRXFIFOStatus(&UART_0))
        {
                DIGITAL_IO_SetOutputHigh(&led1);
                DIGITAL_IO_SetOutputHigh(&led2);
                delay();
                DIGITAL_IO_SetOutputLow(&led1);
                DIGITAL_IO_SetOutputLow(&led2);
                UART_ClearRXFIFOStatus(&UART_0,1);
                car=UART_GetReceivedWord(&UART_0);
                //UART_TransmitWord(&UART_0,car);
                if(car=='1')
                        {
                                if(motor==0)
                                {
                                        PWM_Start(&PWM_0);
                                        motor=1;
                                }
                                duty_cycle=3500;
                                PWM_SetDutyCycle(&PWM_0,duty_cycle);
                                flag=0;
                                DIGITAL_IO_SetOutputHigh(&led1);
                                delay();delay();
                                DIGITAL_IO_SetOutputLow(&led1);
                        }
                if(car=='2')
                {
                        if(motor==0)
                        {
                                PWM_Start(&PWM_0);
                                motor=1;
                        }
                        duty_cycle=4000;
                        PWM_SetDutyCycle(&PWM_0,duty_cycle);
                        flag=1;
                        DIGITAL_IO_SetOutputHigh(&led1);
```

```
                delay();delay();
                DIGITAL_IO_SetOutputLow(&led1);
                                        }
            if(car=='3')
            {
                if(motor==0)
                {
                    PWM_Start(&PWM_0);
                    motor=1;
                }
                duty_cycle=5000;
                PWM_SetDutyCycle(&PWM_0,duty_cycle);
                flag=2;
                DIGITAL_IO_SetOutputHigh(&led1);
                delay();delay();
                DIGITAL_IO_SetOutputLow(&led1);
            }
            if(car=='s')
            {
                PWM_Stop(&PWM_0);
                motor=0;
                flag=0;
            }
        }
    }

}
```

4. Results

Using the knowledge on camshaft and it is principle, a simple Marble Climbing Machine system was created, system which has at its core a complex movement using a single moving part.

Workflow:

- *A marble gets on the first step at the bottom when this step is in its lowest position*
- *When the camshaft rotates the stes position changes.*
- *When the first step reaches its highest position, due to gravity and the design if the step, the marble will get to the second position. This process repeats until the marble reaches the sixth step.*
- *When the marble is on the sixth step and this step reaches its highest position, thanks to the design of the course it will follow and to gravity the marble will reach the starting point and the process repeats*