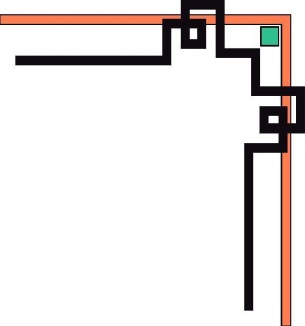
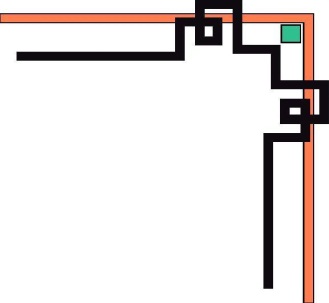
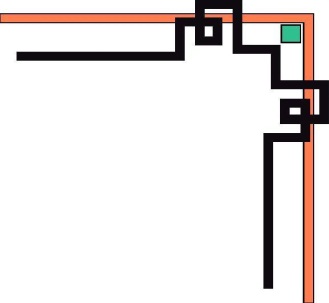
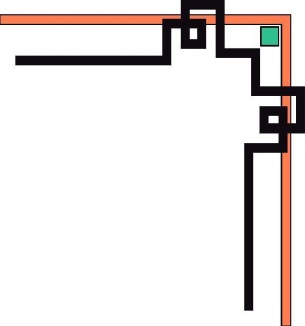
**ĐẠI HỌC QUỐC GIA TP.HCM**



**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA ĐIỆN-ĐIỆN TỬ**

**Bộ môn Điện Tử**

**🙞🙞✰🙜🙜**

Logo, company name

Description automatically generated

**Báo Cáo Bài Tập**

**Điện Tử Ứng Dụng**

*GVHD:* Nguyễn Trung Hiếu

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **MSSV** | **Công việc** |
| Nguyễn Tào Nguyên Phương | 1914744 |  |
| Trần Tài Chung | 1811625 | Mô phỏng winform |
| Dương Nhật Huy | 1810162 | Mô phỏng proteus và báo cáo |
| Võ Văn Hiếu | 1711321 |  |

Nhóm 3

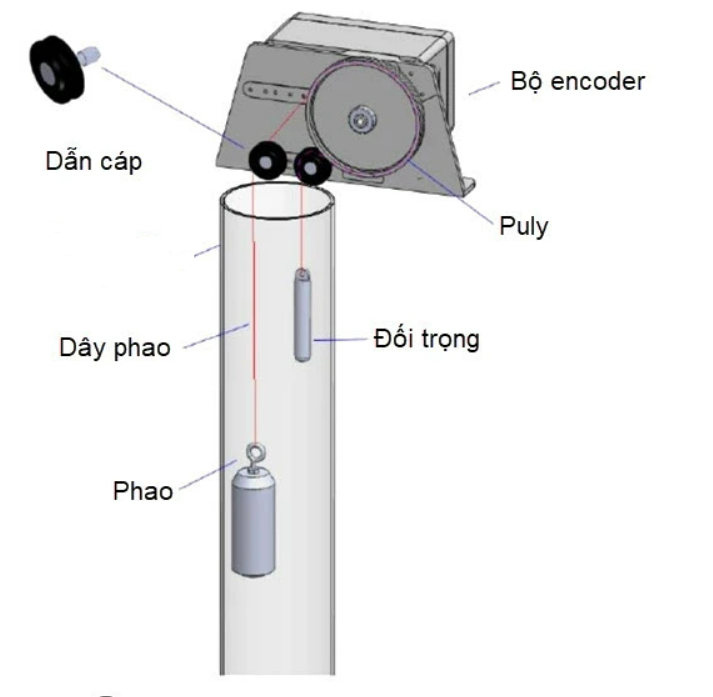
**I – Giới thiệu đề tài**

Trong các ngành công nghiệp hiện nay nói chung hay các ngành chuyên sản xuất, chế tạo nói chung thì việc sử dụng hoặc làm việc với các chất lỏng hay nước là chuyện thường xuyên xảy ra . Cụ thể là trong các nhà máy sản xuất nước giải khát, hay các trạm xử lý nước sinh hoạt,…và còn nhiều lĩnh vực khác nữa.

Chắc chắn trong các lĩnh vực này thì nhu cầu đo lường mực nước hay mực chất lỏng là một điều thiết yếu. Thế nhưng để đo lường trong các thùng chứa, bể chứa cao và lớn thì rất khó khăn. Chính vì đề tài này sẽ giới thiệu một cách để đo và điều khiển mực chất lỏng đó là sử dụng cảm biến phao với encoder

**II – Thiết kế hệ thống**

**1. Đặc tả hệ thống**



**NAME: Thiết bị điều khiển và đo mực nước sử dụng cảm biến phao và Encoder**

**PURPOSE:** Được thiết kế để đo mực nước trong bể, thiết kế chắc chắn của dòng đo mực nước dạng đếm xung kết hợp sử dụng phao và ròng rọc ,tích hợp màn hình LDC cho phép xem giá trị đo bất cứ lúc nào. Nhận mực nước từ 1 thiết bị điều khiển từ đó điều khiển máy bơm để đạt được mức nước yêu cầu

**INPUT/OUTPUT:**

INPUT: Mực nước từ thiết bị điều khiển

OUTPUT: Điều khiển mực nước trong bể đạt tới giá trị yêu cầu. Hiển thị mực nước hiện tại lên LCD

**PERFORMANCE:**

Độ phân giải 1cm

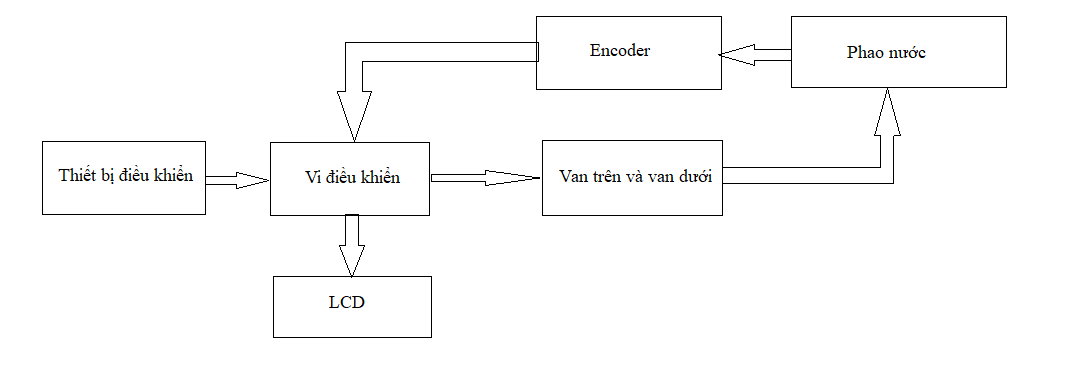
Tầm đo > 3m

Sai số khi đo ± 10 cm

Realtime delay: < 0.1s.

**2 - Sơ đồ phần cứng**

**Sơ đồ khối**

****

# 2.1 Vi điều khiển Arm STM32F103C8T6:

STM32 là một trong những dòng chip phổ biến của ST với nhiều họ thông dụng như F0,F1,F2,F3,F4….. Stm32f103 thuộc họ F1 với lõi là ARM COTEX M3.

STM32F103 là vi điều khiển 32 bit, tốc độ tối đa là 72Mhz. Giá thành cũng khá rẻ so với các loại vi điều khiển có chức năng tương tự. Mạch nạp cũng như công cụ lập trình khá đa dạng và dễ sử dụng.

Một số ứng dụng chính: dùng cho driver để điều khiển ứng dụng, điều khiển ứng dụng thông thường, thiết bị cầm tay và thuốc, máy tính và thiết bị ngoại vi chơi game, GPS cơ bản, các ứng dụng trong công nghiệp, thiết bị lập trình PLC, biến tần, máy in, máy quét, hệ thống cảnh báo, thiết bị liên lạc nội bộ…

A close-up of a computer chip

Description automatically generated with medium confidence

* cổng Mini USB dùng để cấp nguồn, nạp cũng như debug.
* 2 MCU bao gồm 1 MCU nạp và 1 MCU dùng để lập trình.
* Có chân Output riêng cho các chân mạch nạp trên MCU1.
* Có chân Output đầy đủ cho các chân MCU2.
* Chân cấp nguồn ngoài riêng cho MCU2 nếu không sử dụng nguồn từ USB.
* Thạch anh 32,768khz dùng cho RTC và Backup.
* Chân nạp dùng cho chế độ nạp boot loader.
* Nút Reset ngoài và 1 led hiển thị trên chân PB9, 1 led báo nguồn cho MCU2. Board có thể sử dụng một số loại MCU 48 pin của ST(vì sơ đồ chân cũng khá tương đồng giữa các dòng F0, F1, F2, F3, F4) như : STM32F100C8T6, STM32F103C8T6, STM32F303CCT6 ...

# Cấu hình chi tiết của STM32F103C8T6:

ARM 32-bit Cortex M3 với clock max là 72Mhz. Bộ nhớ:

* 64 kbytes bộ nhớ Flash(bộ nhớ lập trình).
* 20kbytes SRAM.

Clock, reset và quản lý nguồn.

* Điện áp hoạt động 2.0V -> 3.6V.
* Power on reset(POR), Power down reset(PDR) và programmable voltage detector (PVD).
* Sử dụng thạch anh ngoài từ 4Mhz -> 20Mhz.
* Thạch anh nội dùng dao động RC ở mode 8Mhz hoặc 40khz.
* Sử dụng thạch anh ngoài 32.768khz được sử dụng cho RTC. Trong trường hợp điện áp thấp:
* Có các mode :ngủ, ngừng hoạt động hoặc hoạt động ở chế độ chờ.
* Cấp nguồn ở chân Vbat bằng pin để hoạt động bộ RTC và sử dụng lưu trữ data khi mất nguồn cấp chính.

2 bộ ADC 12 bit với 9 kênh cho mỗi bộ.

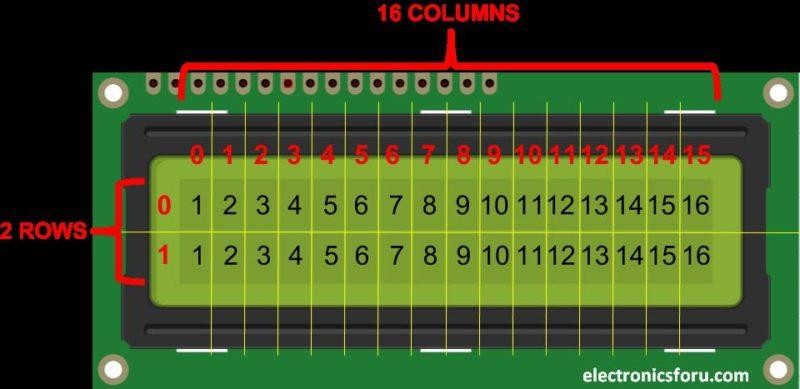
* Khoảng giá trị chuyển đổi từ 0 – 3.6V.
* Lấy mẫu nhiều kênh hoặc 1 kênh.
* Có cảm biến nhiệt độ nội.

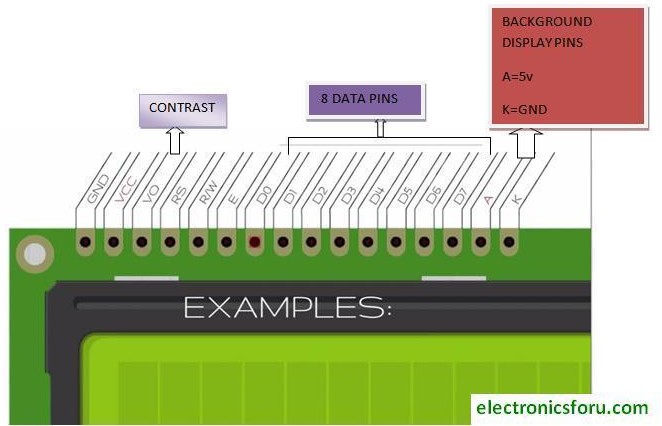
DMA: bộ chuyển đổi này giúp tăng tốc độ xử lý do không có sự can thiệp quá sâu của CPU.

* 7 kênh DMA.
* Hỗ trợ DMA cho ADC, I2C, SPI, UART.

7 timer.

* 3 timer 16 bit hỗ trợ các mode IC/OC/PWM.
* 1 timer 16 bit hỗ trợ để điều khiển động cơ với các mode bảo vệ như ngắt input, dead-time..
* 2 watdog timer dùng để bảo vệ và kiểm tra lỗi.
* 1 sysTick timer 24 bit đếm xuống dùng cho các ứng dụng như hàm Delay…. Hỗ trợ 9 kênh giao tiếp bao gồm:
* 2 bộ I2C(SMBus/PMBus).
* 3 bộ USART(ISO 7816 interface, LIN, IrDA capability, modem control). 2 SPIs (18 Mbit/s).
* 1 bộ CAN interface (2.0B Active) USB 2.0 full-speed interface
* Kiểm tra lỗi CRC và 96-bit ID

**2.2 Màn hình LCD**



|  |  |  |
| --- | --- | --- |
| **Stt** | **Ký hiệu** | **Mô tả** |
| 1 | Vss | Chân nối đất cho LCD, khi thiết kế mạch ta nối chân này với GND của mạch điều khiển |
| 2 | VDD | Chân cấp nguồn cho LCD, khi thiết kế mạch ta nối chân này với VCC=5V của mạch điều khiển |
| 3 | VEE | Điều chỉnh độ tương phản của LCD. |
| 4 | RS | Chân chọn thanh ghi (Register select). Nối chân RS với logic “0” (GND) hoặc logic “1” (VCC) để chọn thanh ghi.  + Logic “0”: Bus DB0-DB7 sẽ nối với thanh ghi lệnh IR của LCD (ở chế độ “ghi” - write) hoặc nối với bộ đếm địa chỉ của LCD (ở chế độ “đọc” - read)  + Logic “1”: Bus DB0-DB7 sẽ nối với thanh ghi dữ liệu DR bên trong LCD. |
| 5 | R/W | Chân chọn chế độ đọc/ghi (Read/Write). Nối chân R/W với logic “0” để LCD hoạt động ở chế độ ghi, hoặc nối với logic “1” để LCD ở chế độ đọc. |
| 6 | E | Chân cho phép (Enable). Sau khi các tín hiệu được đặt lên bus DB0-DB7, các lệnh chỉ được chấp nhận khi có 1 xung cho phép của chân E.  + Ở chế độ ghi: Dữ liệu ở bus sẽ được LCD chuyển vào(chấp nhận) thanh ghi bên trong nó khi phát hiện một xung (high-to-low transition) của tín hiệu chân E.  + Ở chế độ đọc: Dữ liệu sẽ được LCD xuất ra DB0-DB7 khi phát hiện cạnh lên (low-to-high transition) ở chân E và được LCD giữ ở bus đến khi nào chân E xuống mức thấp. |
| 7 -  14 | DB0  - DB7 | Tám đường của bus dữ liệu dùng để trao đổi thông tin với MPU. Có 2 chế độ sử dụng 8 đường bus này :  + Chế độ 8 bit : Dữ liệu được truyền trên cả 8 đường, với bit MSB là bit DB7.  + Chế độ 4 bit : Dữ liệu được truyền trên 4 đường từ DB4 tới DB7, bit MSB là DB7 |
| 15 | A | Nguồn dương cho đèn nền |
| 16 | K | GND cho đèn nền |

**2.3. Encoder:**

**A pair of black headphones on a piece of paper

Description automatically generated with medium confidence**

ENCODER 600/400/100 xung 2 PHA AB / 5-24V

- Trục 6mm

- Dây Red(đỏ) là dây: + (VCC)

- Dây Black(đen) là dây: - (GND)

- Dây White(trắng) là dây: phaseA(xung A)

- Dây Green(xanh) là dây: phaseB(xung B)

2.4 – Puly và sợi cáp

Puly là đĩa quay gắn với encoder . Được khắc lõm để tránh trượt dây cáp gây ảnh hưởng phép đo

A picture containing indoor, music, brass

Description automatically generated A picture containing diagram

Description automatically generated

Còn trên sợi cáp thì ép áp lực mỗi vấu lồi cho chúng bám chạt tương ứng vào đĩa quay.

**2.5 . Phao và đối trọng**

**A close-up of a pen

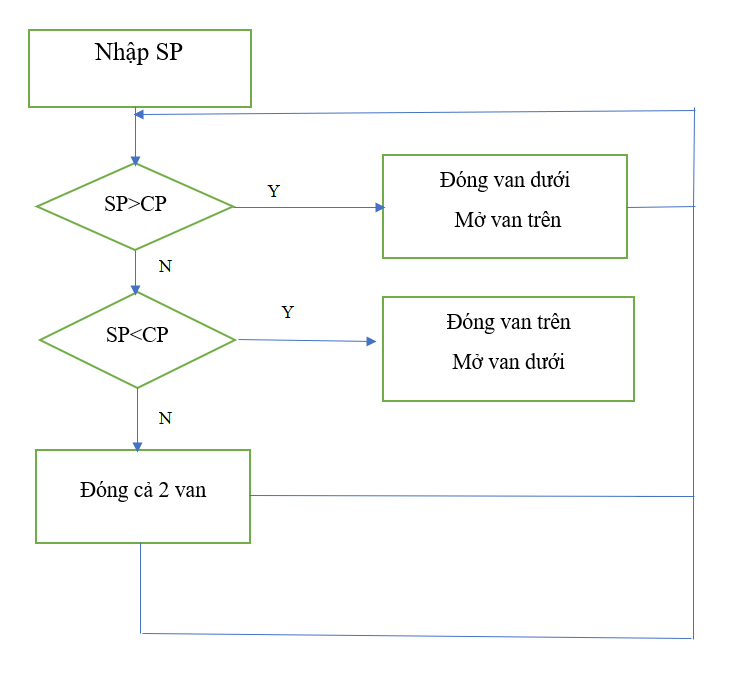
Description automatically generated with low confidence**

Chọn đối trọng nhẹ, đường kính nhỏ, khối lượng lớn tầm 280g.

Phao nặng hơn và nổi trong nước và chế tạo bằng thép không gỉ.

**3. Thiết kế phần mềm và mô phỏng**

**3.1. Lưu đồ giải thuật**

****

Mô tả:

Bước 1 : Nhận giá trị mực nước(SP) từ thiết bị điều khiển gửi đến vi điều khiển. Hiển thị mức nước hiện tại (CV) và mức nước điều khiển (SP) lên LCD

Bước 2 : So sánh với mực nước hiện tại (CV).

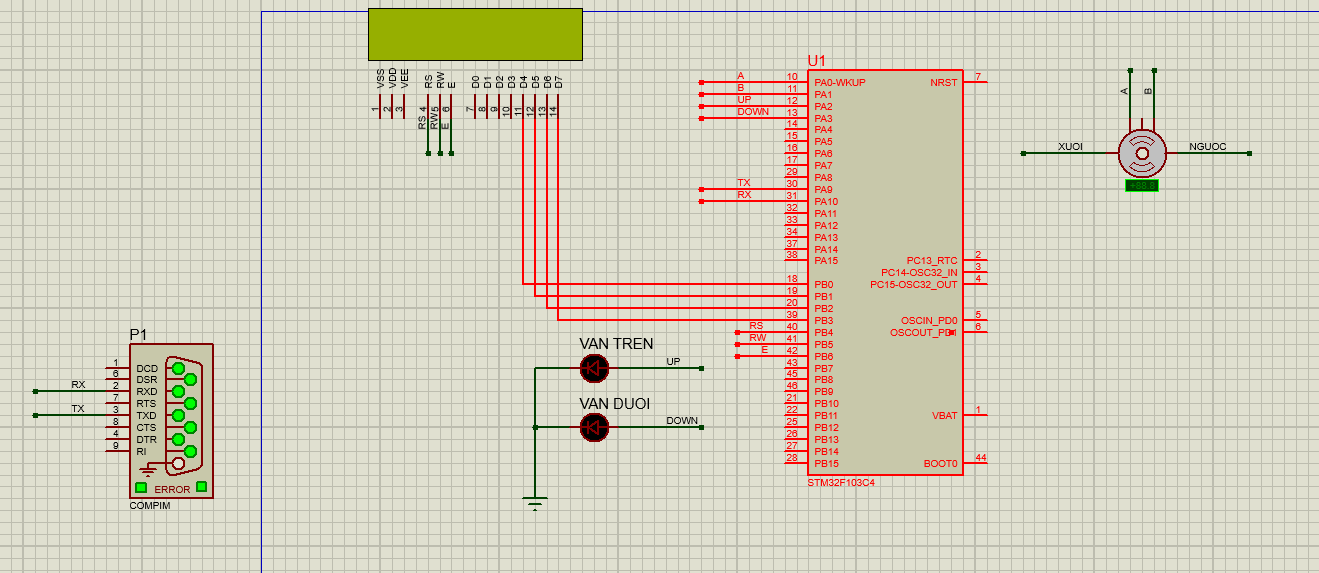
- Nếu SP > CV thì mở van trên,đóng van dưới , bơm nước vào bể

- Nếu SP < CV thì đóng van trên, mở van dưới , xả nước ra khỏi bể

- Nếu SP = CV thì tắt cả 2 van

Bước 3: Quay về bước 1

**3.2 Mô phỏng proteus**

****

- 1 con vi điều khiển thứ 2 dùng để điều khiển encoder nhằm giả lập mực nước

- Xem van trên van dưới được ký hiệu là 2 LED

- Thiết bị điều khiển và hệ thống giao tiếp thông qua UART

**3.3 Mô phỏng Winform**

