Nguyễn Ngọc Khanh

2111474

KSTN L09 TN VXL

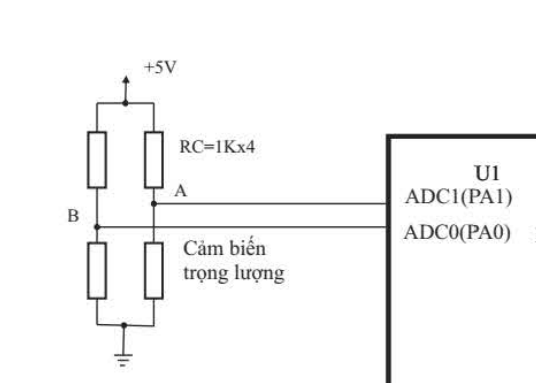
**Trình bày ý tưởng và cách vận hành mô phỏng trên Proteus**

1. **Ý tưởng bài làm**
2. **Cài đặt thông số cho bộ ADC và cài đặt thông số cho cảm biến trọng lượng dạng cầu Wheatstone LOAD CELL để tương thích với nhau.**
3. **ADC**

Chọn ADC mode vi sai tự kích, ngõ vào ADC1 (+), ADC0 (-), độ lợi Gain = 200, Vref = AVcc =5V

1. **LOAD CELL**

- Trong bài làm, em sử dụng cảm biến trọng lượng dạng cầu Wheatstone (LOAD CELL) đóng vai trò như một bàn cân thực tế. Cảm biến hoạt động như sau: khi trọng lượng thay đổi, giá trị biến trở sẽ thay đổi, dẫn đến sự thay đổi về điện áp (do chia áp Vcc) trên biến trở. Kết nối các điện áp trên đầu 2 biến trở (VRa và VRb) (đến ADC1 (+) và ADC0 (-) (VRa kết nối với ADC1, VRb kết nối với ADC0) để đọc sự chênh lệch điện áp, chuyển thành giá trị ADC (VRa – VRb = Vab là sự chênh lệch điện áp mà bộ ADC vi sai đọc).



- Vì số kg đọc từ bộ ADC, mà Dout của bộ ADC mode vi sai chỉ có 500 (thực tế là 512, nhưng ta chỉ đọc tối đa 10kg là số kg lớn nhất của trường hợp mít già, 10:2 = 5 tương đương Dout = 500 nên ta không xét từ 501 đến 512) nên khi ADC chuyển đổi xong, ta phải nhân giá trị ADC với 2.

**-** Tính toán các thông số cho LOAD CELL:

**+** Số kg tối đa mà LOAD CELL cân là 10kg (số kg tối đa của trường hợp mít già), từ 10.01kg đến 10.24kg (ADC x 2 mang giá trị 1001 đến 1024) không được phân loại là loại mít nào nên ta bỏ qua.

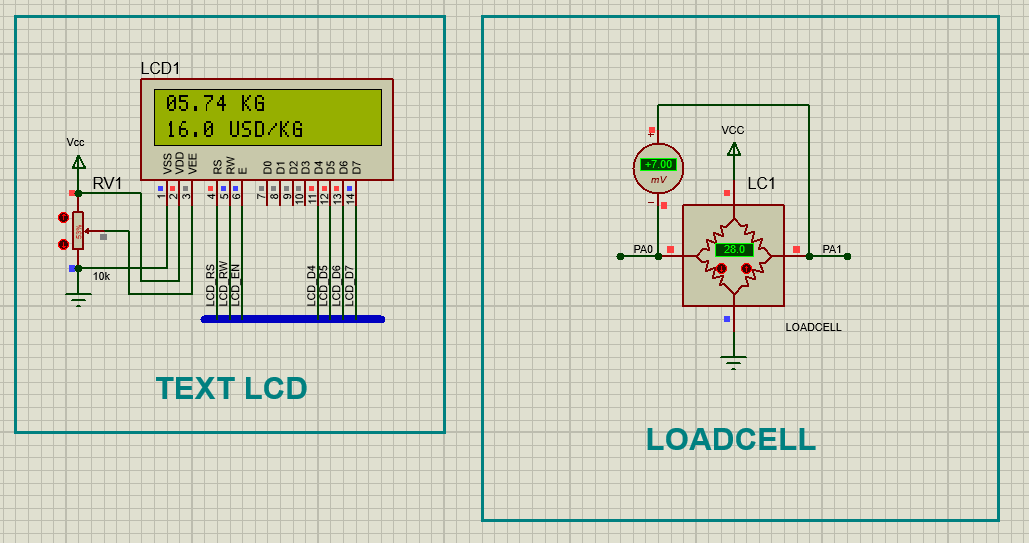
**+** Điện áp cấp vào cảm biến là Vcc = 5V, điện trở nội của cảm biến ở trạng thái tĩnh trên mỗi nhánh là 1000 ohm.

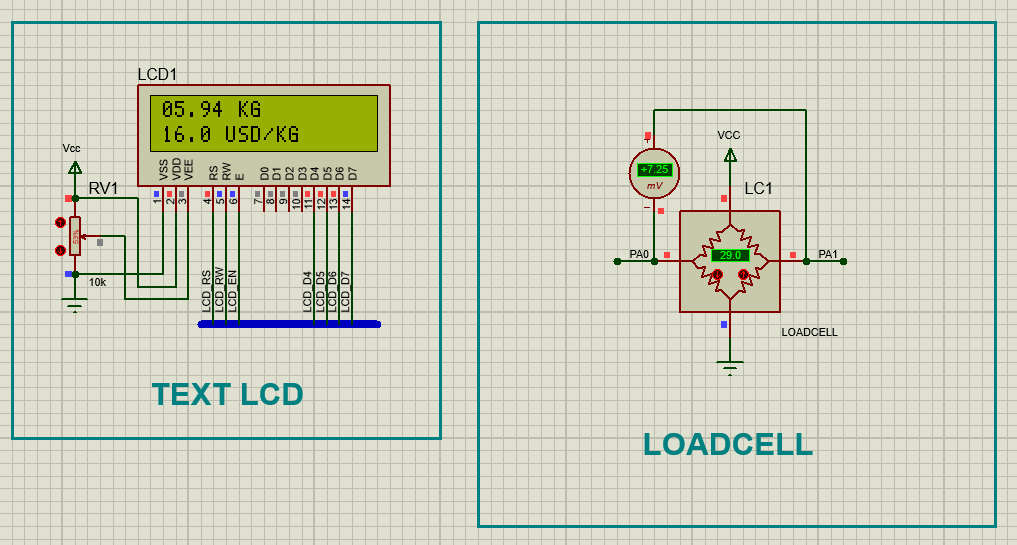
**+** Điện áp ngõ ra cảm biến: Vabmax = (VRa – VRb) max = FS x Vcc = FS x 5 (V)

**+** Vref = ****suy ra 

**+** Vậy Vabmax = FS x 5 = 25mV suy ra độ nhạy toàn tầm FS = Vabmax/5 = (25mV)/(5V) = 5mV/V (độ nhạy toàn tầm FS điều chỉnh được, điều chỉnh trong Proteus).

+ Khi thay đổi 1 đơn vị trên LOAD CELL (VD: 28.0 lên 29.0) thì số kg thay đổi 0.2 kg (05.74 kg lên 05.94 kg tương ứng với 28.0 lên 29.0).





1. **Một số kí hiệu dùng trong chương trình**
2. **Về đơn giá thực tế:**

- Đơn giá thực tế là các giá trị như 12.5 USD/KG, 16.0 USD/KG.

- Thanh ghi DON\_GIA (R18) là thanh ghi chứa giá trị (đơn giá thực tế x10), ví dụ như đơn giá mít là 12.5 USD/KG thì DON\_GIA = 12.5 x 10 = 125.

1. **Về số KG thực tế:**

- Số KG thực tế là các giá trị như 05.57 KG, 06.32 KG

- Số KG thực tế có được bằng cách chèn dấu chấm thập phân vào giữa 4 digits của số BCD của giá trị (ADC x 2) sao cho có 2 số thập phân sau dấu chấm. Ví dụ như (ADC x 2) = 0557 thì số KG thực tế bằng 05.57 KG. Hay nói cách khác, giá trị (ADC x 2) = (số KG thực tế) x 100.

1. **Về giá tiền thực tế:**

- Giá tiền thực tế là số tiền thu được khi lấy đơn giá thực tế nhân với số KG thực tế. Ví dụ đơn giá thực tế = 10.0 USD/KG, số KG thực tế là 1.5 KG (mít non) thì giá tiền thực tế = 10.0 x 1.5 = 15 USD.

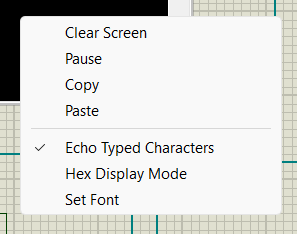
- Trong chương trình sử dụng 3 thanh ghi GIA\_TIEN (GIA\_TIEN\_HIGH: GIA\_TIEN\_MIDDLE: GIA\_TIEN\_LOW) (R25: R24: R23). GIA\_TIEN là số nhị phân 24 bit, là số BCD 6 digits.

- Giá trị nằm trong 24 bit của 3 thanh ghi GIA\_TIEN = (giá tiền thực tế) x1000 = (ADC x 2) x DON\_GIA = (ADC x 2) x (đơn giá thực tế) x 10 = (số KG thực tế) x 100 x (đơn giá thực tế) x 10.

- Giá tiền thực tế có được bằng cách đặt dấu chấm thập phân vào giữa 6 số BCD của giá trị thanh ghi GIA\_TIEN. Ví dụ như GIA\_TIEN = 112523 (số thập phân) thì giá tiền thực tế bằng 112.523 USD. Ta có được điều này vì GIA\_TIEN = (giá tiền thực tế) x1000.

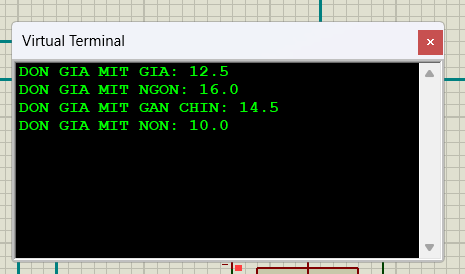
1. **Ý tưởng bài làm**
2. **Viết chương trình cho người dùng nhập đơn giá cùa 4 loại mít lần lượt là mít già, mít ngon, mít gần chín và mít non (yêu cầu của câu c)**

- Trước khi nhập, chọn “Echo Typed Characters” để hiện lên những kí tự mà người dùng sẽ nhập (kiểm tra xem kí tự mình nhập có đúng với mong muốn hay không).



- Đơn giá mỗi loại mít cần nhập đúng (nếu không thì phải stop và run lại trên Proteus và nhập lại) và đủ 4 kí tự gồm 3 số BCD và 1 dấu chấm thập phân. Ví dụ như khi hiện lên dòng chữ “DON GIA MIT GIA: ’’ thì người dùng nhập 4 kí tự, chẳng hạn như “12.5’’, 4 kí tự này gồm ‘1’, ‘2’, ‘.’, ‘5’. Lần lượt như vậy cho đến khi nhập đủ đơn giá của 4 loại mít (16 kí tự).

- Trước khi lưu 4 kí tự của đơn giá loại mít nào đó, luôn xuất hiện dòng chữ DON GIA MIT …, sau khi nhập đủ 4 kí tự sẽ xuất ra mã ASCII 0x0D để xuống dòng nhập đơn giá loại mít tiếp theo, tiếp tục hiện dòng chữ phía trên để mời người dùng nhập tiếp.



- Lưu lần lượt 16 kí tự này vào các địa chỉ SRAM bắt đầu từ 0x260 đến 0x26F, cụ thể:

+ Trước khi lưu, ta chuyển sẵn mã ASCII nhận được từ Hercules truyền xuống thành số BCD bằng cách ANDI với 0x0F để xóa 4 bit cao, giữ 4 bit thấp. Ví dụ như ta nhận được mã ASCII của số 2 là 0x32, ta ANDI 0x32 với 0x0F ta sẽ thu được 0x02 là BCD của số 2.

+ Từ địa chỉ 0x260 đến 0x263 lưu 4 kí tự của đơn giá mít già. VD: người dùng nhập 4 kí tự, chẳng hạn như “12.5’’ thì ‘1’ sẽ lưu ở 0x260, ‘2’ sẽ lưu ở 0x261, ‘.’ sẽ lưu ở 0x262 và ‘5’ sẽ lưu ở 0x263.

+ Từ địa chỉ 0x264 đến 0x267 lưu 4 kí tự của đơn giá mít ngon.

+ Từ địa chỉ 0x268 đến 0x26B lưu 4 kí tự của đơn giá mít gần chín.

+ Từ địa chỉ 0x26C đến 0x26F lưu 4 kí tự của đơn giá mít non.

1. **Viết chương trình con tính giá trị của thanh ghi DON\_GIA trong 4 trường hợp của mít**

- Sau khi nhập đủ các đơn giá của 4 loại mít, viết 4 chương trình con tính giá trị của DON\_GIA = (đơn giá thực tế x10) bằng cách như sau:

- DON\_GIA = (BCD hàng trăm) x 100 + (BCD hàng chục) x 10 + BCD hàng đơn vị.

- Ta bỏ qua địa chỉ chứa dấu chấm thập phân.

- Ví dụ như trong chương trình con tính giá mít ngon:

+ Lấy BCD của chữ số hàng trăm được lưu ở địa chỉ 0x264 ta nhân với 100, kết quả được lưu ở R0 thì ta cộng thanh ghi DON\_GIA với R0.

+ Lấy BCD của chữ số hàng chục được lưu ở địa chỉ 0x265 ta nhân với 10, kết quả được lưu ở R0 thì ta cộng thanh ghi DON\_GIA với R0.

+ Dấu chấm thập phân được lưu ở 0x266, ta bỏ qua.

+ Lấy BCD của chữ số hàng đơn vị được lưu ở địa chỉ 0x267, cộng DON\_GIA với BCD hàng đơn vị.

+ VD: 0x264 đến 0x267 lần lượt lưu số 1, số 2, dấu chấm và số 5. DON\_GIA = 1x100 + 2x10 + 5 = 125.

1. **Viết chương trình con tách DON\_GIA thành BCD 3 digits**

- Sau khi có đủ 4 giá trị DON\_GIA ở 4 trường hợp mít khác nhau, ta viết chương trình con tách DON\_GIA lại thành số BCD 3 digits, chuyển 3 digits này thành mã ASCII để chuẩn bị xuất ra LCD. Sau đó lưu mã ASCII của 2 digits nằm trước dấu chấm thập phân ở 2 vị trí đầu, vị trí thứ 3 lưu dấu chấm thập phân, vị trí thứ 4 lưu mã ASCII của 1 digit nằm sau dấu chấm thập phân. Như vậy khi xuất ra LCD ta sẽ xuất được đơn giá thực tế. Cụ thể:

+ Địa chỉ SRAM từ 0x20B (USD\_PER\_KG\_ADR + 11) đến 0x200 (USD\_PER\_KG\_ADR) sẽ lưu chuỗi kí tự “EF.G USD/KG” và kí tự kết thúc (0x00) để LCD nhận biết khi nào kết thúc. Với EFG là giá trị thanh ghi DON\_GIA và EF.G là đơn giá thực tế.

+ Ví dụ như EF.G = 14.5 thì: 0x20B lưu kí tự ‘1’, 0x20A lưu kí tự ‘4’, 0x209 lưu kí tự ‘.’, …, 0x200 lưu kí tự 0x00 (kết thúc).

1. **Viết chương trình phân loại mít dựa vào số KG đọc được.**

- Tùy thuộc vào mít thuộc loại nào mà sẽ có DON\_GIA khác nhau, tính tiền cũng sẽ khác nhau.

- Ta thực hiện việc đọc giá trị ADC, sau đó nhân 2 giá trị ADC (do ADC mode vi sai tối đa chỉ đọc được +512), ta sẽ thu được (ADC x 2) tương ứng với số KG được cân trên LOAD CELL.

- (ADC x 2) là số binary 10 bit, ta so sánh các trường hợp của 2 bit cao trước, 8 bit thấp sau, cụ thể

+ Nếu (bit 10 = 1) và (bit 9 = 1), giá trị (ADC x 2) luôn lớn hơn hoặc bằng 768, hay số kg thực tế luôn lớn hơn hoặc bằng 7.68 kg, luôn là mít già.

+ Nếu (bit 10 = 1) và (bit 9 = 0), giá trị (ADC x 2) luôn lớn hơn hoặc bằng 512, hay số kg thực tế luôn lớn hơn hoặc bằng 5.12 kg, là mít già hoặc mít ngon.

++ Nếu 8 bit thấp bé hơn 88, hay (ADC x 2) < 600, số kg thực tế bé hơn 6kg

Là mít ngon

++ Ngược lại là mít già

+ Nếu (bit 10 = 0) và (bit 9 = 1), giá trị (ADC x 2) luôn lớn hơn hoặc bằng 256, hay số kg thực tế luôn lớn hơn hoặc bằng 2.56 kg, là mít ngon hoặc mít gần chín.

++ Nếu 8 bit thấp bé hơn 144, hay (ADC x 2) < 400, số kg thực tế bé hơn 4kg là mít gần chín

++ Ngược lại là mít ngon

+ Nếu (bit 10 = 0) và (bit 9 = 0), giá trị (ADC x 2) luôn nhỏ hơn hoặc bằng 255, hay số kg thực tế luôn nhỏ hơn hoặc bằng 2.55 kg, là mít gần chín hoặc mít non.

++ Nếu 8 bit thấp bé hơn 200, hay (ADC x 2) < 200, số kg thực tế bé hơn 2kg là mít non

++ Ngược lại là mít gần chín

1. **Viết chương trình con tách (ADC x 2) thành BCD 4 digits**

- Viết chương trình con tách 10 bit giá trị (ADC x 2) thành số BCD 4 digits, chuyển 4 digits này thành mã ASCII để chuẩn bị xuất ra LCD. Sau đó lưu mã ASCII của 2 digits nằm trước dấu chấm thập phân ở 2 vị trí đầu, vị trí thứ 3 lưu dấu chấm thập phân, vị trí thứ 4 và 5 lưu mã ASCII của 2 digits nằm sau dấu chấm thập phân. Như vậy khi xuất ra LCD ta sẽ xuất được số KG thực tế. Cụ thể:

- Địa chỉ SRAM từ 0x214 (KG\_ADR + 8) đến 0x20C (KG\_ADR) sẽ lưu chuỗi kí tự “AB.CD KG” và kí tự xuống dòng (0x0D) để LCD nhận biết khi nào xuống dòng. Với ABCD là giá trị (ADC x 2) và AB.CD là số KG thực tế.

- Ví dụ như AB.CD = 05.57 thì: 0x214 lưu kí tự ‘0’, 0x213 lưu kí tự ‘5’, 0x212 lưu kí tự ‘.’, …, 0x20C lưu kí tự 0x0D (xuống dòng).

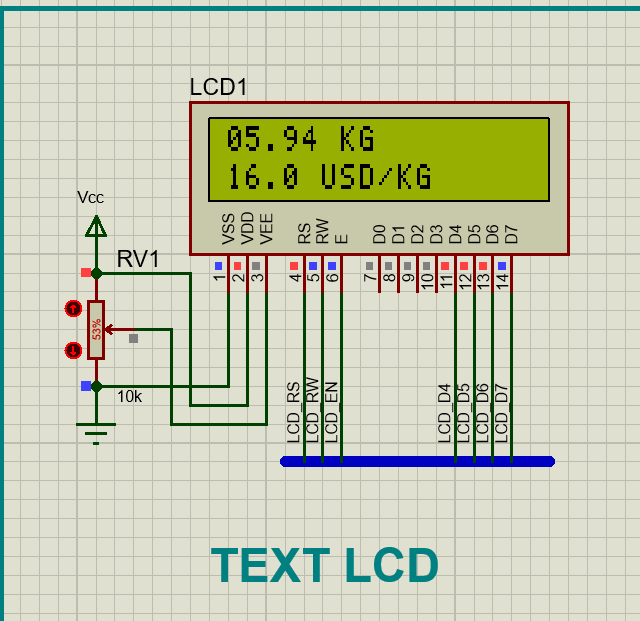
1. **Viết chương trình ngắt ngoài INT0 tác động cạnh xuống, nối với SW0. Nhấn SW0 xuất ra số kg và đơn giá thực tế lên LCD.**

Chương trình ngắt INT0 tác động cạnh xuống sẽ xuất kí tự từ địa chị SRAM bắt đầu từ 0x214 đến 0x200 khi SW0 được nhấn, trong đó:

+Từ 0x214 đến 0x20C là dòng 1 của LCD, xuất số kg của mít và kí tự xuống dòng.

**+** Từ 0x20B đến 0x200 là dòng 2 của LCD, xuất đơn giá thực tế của mít và kí tự kết thúc

**+** 0x20C và 0x20B nối liền địa chỉ của 2 dòng xuất lên LCD để thuận tiện cho việc đọc.



1. **Viết chương trình con tính tiền**

Chương trình con tính tiền thực hiện nhiệm vụ nhân (ADC x 2) với DON\_GIA, trong đó (ADC x 2) là số nhị phân 10 bit, DON\_GIA là số nhị phân 8 bit.

Kết quả thu được là 24 bit nằm trong 3 thanh ghi GIA\_TIEN\_HIGH: GIA\_TIEN\_MIDDLE: GIA\_TIEN\_LOW

1. **Viết chương trình con tách GIA\_TIEN thành số BCD 6 digits**

- Viết chương trình con tách 24 bit GIA\_TIEN thành số BCD 6 digits, chuyển 6 digits này thành mã ASCII để chuẩn bị xuất ra LCD.

- Ta chèn dấu chấm thập phân vào giữa 6 digits để có được giá tiền thực tế sao cho sau dấu chấm thập phân có 3 số thập phân.

- Lưu vào địa chỉ SRAM như sau: bắt đầu từ 0x252 đến 0x240 lưu lần lượt các kí tự của dòng chữ “THANH TIEN”, 0x0D, “HIJ.KLM”, 0x00. Tổng cộng gồm 19 kí tự bao gồm cả khoảng trắng, dấu chấm thập phân, kí tự xuống dòng và kí tự kết thúc. Trong đó, HIJKLM là giá trị GIA\_TIEN 24 bit, HIJ.KLM là giá tiền thực tế (đã được đặt dấu chấm thập phân đúng chỗ).

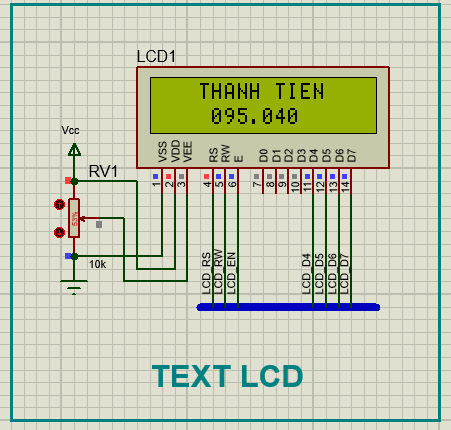
1. **Viết chương trình ngắt ngoài INT1 tác động cạnh xuống, nối với SW1. Nhấn SW1 xuất ra dòng chữ “THANH TIEN” và giá tiền thực tế lên LCD.**

Chương trình ngắt INT1 tác động cạnh xuống sẽ xuất kí tự từ địa chị SRAM bắt đầu từ 0x252 đến 0x240 khi SW1 được nhấn, trong đó:

+Từ 0x252 đến 0x248 là dòng 1 của LCD, xuất “THANH TIEN” và kí tự xuống dòng.

**+** Từ 0x247 đến 0x240 là dòng 2 của LCD, xuất giá tiền thực tế của mít và kí tự kết thúc.

**+** 0x248 và 0x247 nối liền địa chỉ của 2 dòng xuất lên LCD để thuận tiện cho việc đọc.

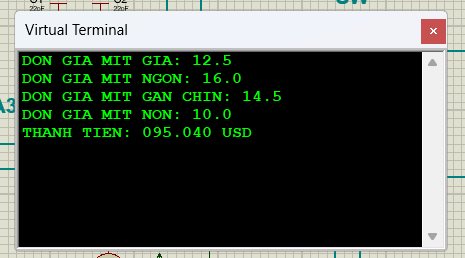


1. **Viết chương trình ngắt ngoài INT2 tác động cạnh xuống, nối với SW2. Nhấn SW2 xuất ra dòng chữ “THANH TIEN: HIJ.KLM USD” lên màn hình máy tính, với HIJ.KLM là giá tiền thực tế.**

Chương trình ngắt INT2 tác động cạnh xuống sẽ xuất từng kí tự tạo thành chuỗi kí tự “THANH TIEN: HIJ.KLM”khi SW2 được nhấn, trong đó:

+ “THANH TIEN: ……. USD’’ được xuất cố định lên Hercules.

+ HIJ.KLM được đọc từ địa chỉ SRAM từ 0x247 đến 0x241, đọc lần lượt từng kí tự và xuất lên Hercules



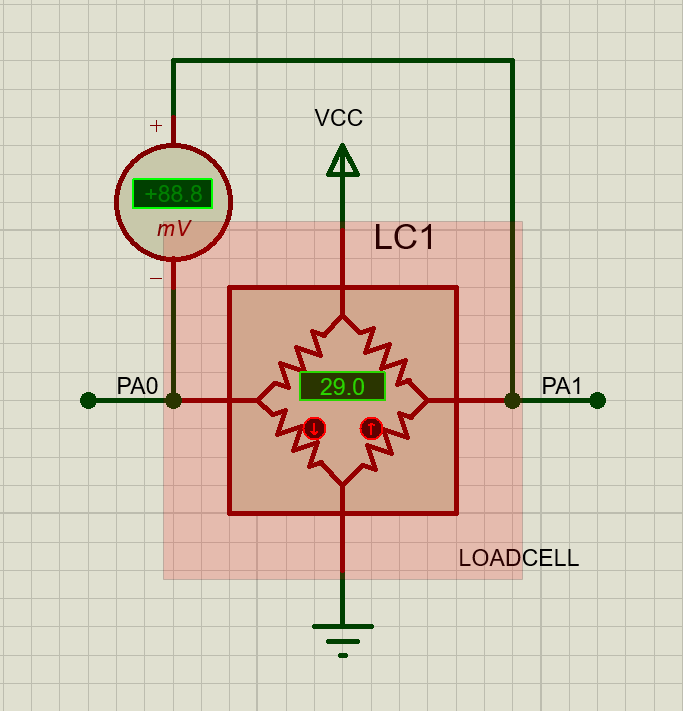
1. **Lưu ý**

Các chương trình ngắt ngoài xuất lên LCD số kg và đơn giá (INT0), giá tiền (INT1), xuất lên Hercules giá tiền (INT2) được đáp ứng ngay lập tức. Do đó khi đã xuất lên LCD và Hercules mà có sự thay đổi số kg trên LOAD CELL, người dùng phải nhấn lại SW tương ứng để có được kết quả chính xác.

1. **Cách vận hành mô phỏng Proteus**
2. **Chỉnh khối lượng trên LOAD CELL**

- LOAD CELL có 2 nút tăng hoặc giảm số (lưu ý: số này không bằng giá trị khối lượng đọc được ở ADC x 2). Khi người dùng bấm, số sẽ thay đổi 1 step (step này chỉnh được). Khi số tăng hoặc giảm thì ADC cũng sẽ tăng hoặc giảm tương ứng, dẫn đến khối lượng cũng tăng hoặc giảm tương ứng.

- Ở trong mô phỏng, step đang bằng 1 đơn vị (mỗi lần nhấn sẽ thay đổi 1 đơn vị), mỗi step sẽ làm thay đổi 0.2 kg.

****

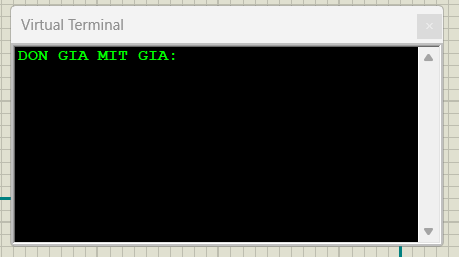
1. **Nhập đơn giá 4 loại mít**

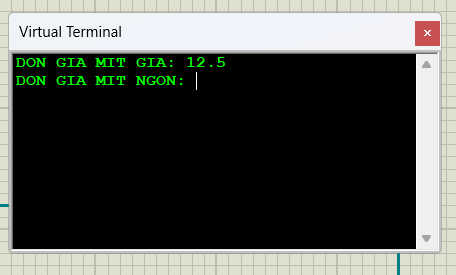
**-** Khi bắt đầu chạy mô phỏng, sẽ có cửa sổ giao tiếp mời người dùng nhập lần lượt đơn giá của từng loại mít cho đến khi nhập đủ 4 loại mít.

**-** Trước khi nhập đơn giá, người dùng phải click chuột phải, nhấn chọn “Echo Typed Characters” để xem được nội dung mình nhập.

**-** Người dùng cần nhập đúng và đủ 4 kí tự, gồm 3 số và 1 dấu chấm. VD: đơn giá là 12.5 thì người dùng nhập: ‘1’, ‘2’, ‘.’, ‘5’.

**-** Khi nhập đủ 4 kí tự sẽ tự động xuống dòng và mời nhập đơn giá của loại mít khác.





1. **Các nút nhấn**

**-** INT0 tác động cạnh xuống, nối với SW0. Nhấn SW0 sẽ xuất ra số kg và đơn giá thực tế lên LCD.

**-** INT1 tác động cạnh xuống, nối với SW1. Nhấn SW1 sẽ xuất ra giá tiền thực tế lên LCD.

**-** INT2 tác động cạnh xuống, nối với SW2. Nhấn SW2 sẽ xuất ra giá tiền thực tế lên màn hình máy tính

**-** Các chương trình ngắt ngoài xuất lên LCD số kg và đơn giá (INT0), giá tiền (INT1), xuất lên Hercules giá tiền (INT2) được đáp ứng ngay lập tức. Do đó khi đã xuất lên LCD và Hercules mà có sự thay đổi số kg trên LOAD CELL, người dùng phải nhấn lại SW tương ứng để có được kết quả chính xác.