**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**A blue logo with a flower

Description automatically generated**

**BÁO CÁO THỰC HÀNH MÔN VI XỬ LÝ – VI ĐIỀU KHIỂN**

**BÀI THỰC HÀNH SỐ 6: SỬ DỤNG UART**

***Sinh viên thực hiện:***  
Trần Ngọc Ánh 22520077

***Giảng viên hướng dẫn:*** Phạm Minh Quân

***Mã lớp:*** CE103.O22

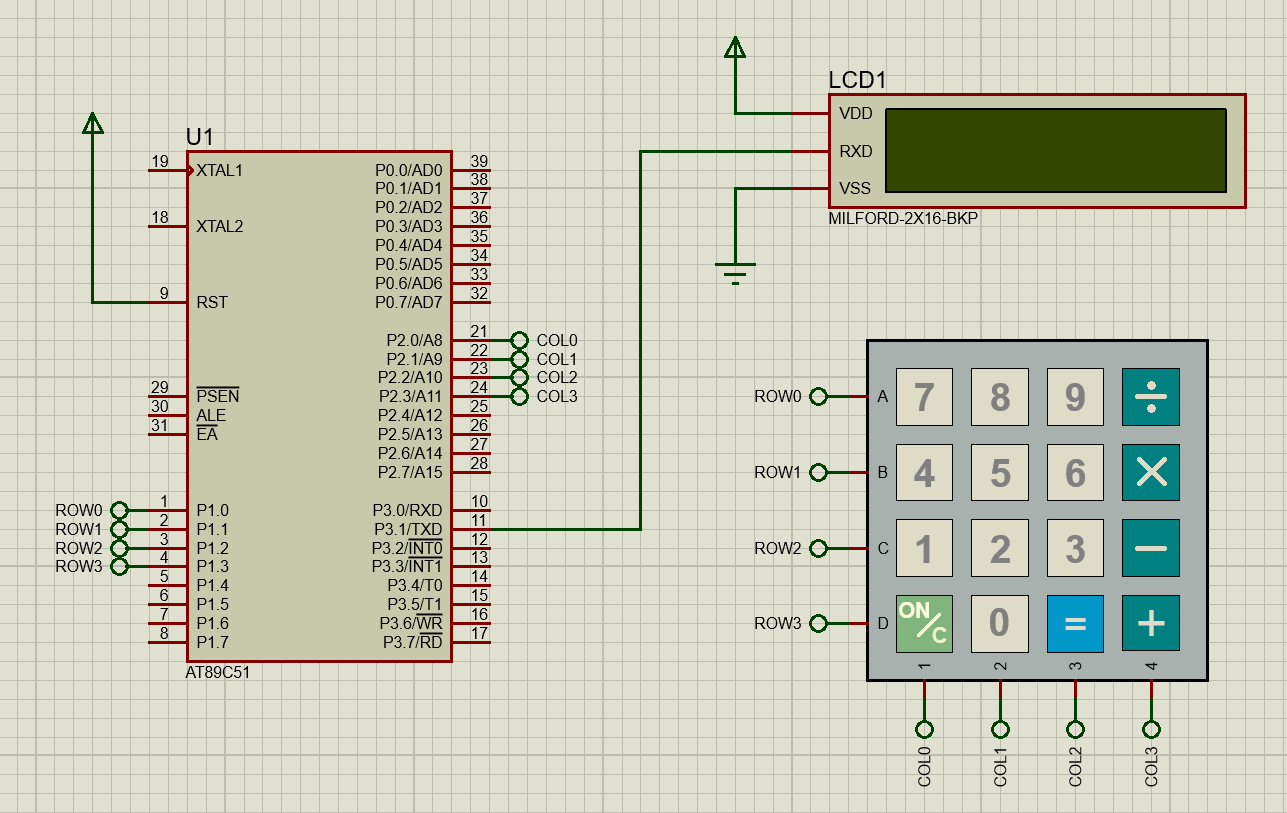
**TP. HỒ CHÍ MINH, 30 THÁNG 5 NĂM 2024**

**Yêu cầu thực hành:**

1. **Xây dựng bộ bàn phim 4x4 gồm các nút sau:**

* Từ 0 🡪 9
* Các dấu +, -, \*, /
* Dấu =
* Nút Reset

1. **Sử dụng AT89C51/AT89C52 kết hợp với module vừa thiết kế trên để thiết kế một máy tính cầm tay, hiển thị phép tính và kết quả trên 1 LCD nhận dữ liệu qua UART.**
2. ***Mô phỏng thiết kế:***



1. ***Source code và giải thích nguyên lí:***

|  |  |
| --- | --- |
| **Code** | **Nguyên lí hoạt động** |
| N1 EQU 30H  N2 EQU 31H  OP EQU 32H  R EQU 33H  SIGN EQU 34H  TEMP EQU 35H  DIF BIT 0AH  OIF BIT 0BH  AIF BIT 0CH  Space DATA 32d  ; Reset Vector  org 0000h  jmp Start  ; Code Segment  ; Main Program  org 0100h  Start:  MOV TMOD, #021h  LCALL LCDInit  MOV R5,Space  Loop:  LCALL KTdchnho  LCALL Nhan  LCALL Tinhtoan  LCALL IN  JMP Loop  LCDInit:  MOV TH1, #0FDh  MOV SCON, #50h  SETB TR1  RET  KTdchnho:  CLR A  MOV N1,A  MOV N2,A  MOV OP,A  MOV R,A  MOV SIGN,A  MOV TEMP,A  CLR C  MOV DIF,C  MOV OIF,C  MOV AIF,C  RET  Nhan:  ACALL docnutnhan  ACALL XL\_input  ACALL LCD\_hien  DEC R5  ANL A,#0FH  MOV N1,A    ACALL docnutnhan  ACALL XL\_input  ACALL LCD\_hien  DEC R5  MOV OP,A  ACALL docnutnhan  ACALL XL\_input  ACALL LCD\_hien  DEC R5  ANL A,#0FH  MOV N2,A  ACALL docnutnhan  ACALL XL\_input  ACALL LCD\_hien  DEC R5  RET  docnutnhan:  MOV P1,#0FH  K1:  MOV P2,#0 MOV A,P1  ANL A,#00001111B  CJNE A,#00001111B,K1  K2:  LCALL DELAY  MOV A,P1  ANL A,#00001111B  CJNE A,#00001111B,Doco  SJMP K2  Doco:  MOV P2,#11111110B MOV A,P1  ANL A,#00001111B  CJNE A,#00001111B,ROW\_0  MOV P2,#11111101B  MOV A,P1  ANL A,#00001111B  CJNE A,#00001111B,ROW\_1  MOV P2,#11111011B  MOV A,P1  ANL A,#00001111B  CJNE A,#00001111B,ROW\_2  MOV P2,#11110111B  MOV A,P1  ANL A,#00001111B  CJNE A,#00001111B,ROW\_3  LJMP K2  ROW\_0:  MOV DPTR,#KCODE0  SJMP FIND  ROW\_1:  MOV DPTR,#KCODE1  SJMP FIND  ROW\_2:  MOV DPTR,#KCODE2  SJMP FIND  ROW\_3:  MOV DPTR,#KCODE3  FIND:  RRC A  JNC MATCH  INC DPTR  SJMP FIND  MATCH:  CLR A  MOVC A,@A+DPTR  JZ ON  RET  ON:  ACALL Xoa  JMP Start  XL\_input:  CJNE A,#"+", next1  AJMP found  next1:  CJNE A,#"-", next2  AJMP found  next2:  CJNE A,#"\*", next3  AJMP found  next3:  CJNE A,#"/", next4  AJMP found  next4:  CJNE A,#"=", next5  CLR DIF  CLR OIF  SETB AIF  RET  next5:  SETB DIF  CLR OIF  CLR AIF  RET  found:  CLR DIF  SETB OIF  CLR AIF  RET  Tinhtoan:  MOV A,N1  MOV B,N2  MOV R7,OP  CJNE R7,#"+",NEXT11  ADD A,B  MOV R,A  MOV SIGN,#"+"  RET  NEXT11:  CJNE R7,#"-",NEXT22  SUBB A,B  JC Truam  MOV R,A  MOV SIGN,#"+"  RET  Truam:  CPL A  INC A  MOV R,A  MOV SIGN,#"-"  RET  NEXT22:  CJNE R7,#"\*",NEXT33  MUL AB  MOV R,A  MOV SIGN,#"+"  RET  NEXT33:  CJNE R7,#"/",NEXT44  DIV AB  MOV R,A  MOV SIGN,#"+"  NEXT44:  RET  IN:  MOV R6,SIGN  CJNE R6,#"+",SINGED  RETURN:  MOV A,R  MOV B,#0AH  DIV AB  JZ LESSTEN  ADD A,#30h  ACALL LCD\_hien  DEC R5  MOV A,B  ADD A,#30h  ACALL LCD\_hien  DEC R5  JMP DONE  LESSTEN:  MOV A,B  ADD A,#30h  ACALL LCD\_hien  DEC R5  JMP DONE  SINGED:  MOV A,#"-"  ACALL LCD\_hien  DEC R5  JMP RETURN  DONE:  RET  Xoa:  MOV A,#" "  ResetVT:  ACALL LCD\_hien  DJNZ R5,ResetVT  MOV R5,Space  Clear:  ACALL LCD\_hien  DJNZ R5,Clear  MOV R5,Space  RET  LCD\_hien:  MOV SBUF,A  Wait:  JNB TI, Wait  CLR TI  RET  DELAY:  LOO:  MOV TL0, #LOW(-20000)  MOV TH0, #HIGH(-20000)  SETB TR0  LOOP2:  JNB TF0, LOOP2  CLR TR0  CLR TF0  RET  LDELAY:  LLOP:  MOV TL0, #LOW(-10000)  MOV TH0, #HIGH(-10000)  SETB TR0  LLOOP2:  JNB TF0, LLOOP2  CLR TR0  CLR TF0  RET  ORG 300H  KCODE0: DB "7","4","1",0  KCODE1: DB "8","5","2","0"  KCODE2: DB "9","6","3","="  KCODE3: DB "/","\*","-","+"  End | Địa chỉ lưu toán hạng đầu tiên  Địa chỉ lưu toán hạng thứ hai  Địa chỉ lưu dấu  Địa chỉ lưu kết quả  Biến SIGN để xác định số xuất ra là âm hay dương  Thể hiện số  Thể hiện các ký tự +, -, \*, /  Thể hiện dấu “=”  Hàm *KTdchnho* đảm bảo các thanh ghi đang lưu kết quả là 0.  Hàm *Nhan* nhận tín hiệu của phép tính.  Đọc nút nhấn.  Xử lý các nhóm tín hiệu.  Đọc toán hạng đầu tiên.  Đọc dấu của phép tính.  Đọc toán hạng thứ hai.  Đọc dấu bằng.  Hàm *K2* giúp chống đợi phím khi nhấn.  Hàm delay để tránh việc phím nhấn nhiều lần khi ta nhấn.  Dò theo từng cột để xác nhận cột đang được nhấn.  4 hàm *ROW\_0, ROW\_1, ROW\_2, ROW\_3* sẽ đại diện 4 hàng trỏ đến ký tự tương ứng.  Hàm kiểm tra ký tự nhập vào thuộc nhóm nào trong 3 nhóm.  Nếu là dấu “=” thì sẽ nhảy đến hàm này, set DIF lên 1.  Nếu là dấu +, -, \*, / thì sẽ nhảy đến hàm này, set OIF lên 1.  Hàm tính toán các giá trị:  - Hai toán hạng sẽ được chuyển từ N1, N2 đến hai thanh ghi A, B.  - Dấu sẽ chuyển từ OP sang R7.  Phép cộng 2 số.  Phép trừ 2 số.  Nhảy đến *Truam* khi c = 1 (phép trừ trên ra kết quả âm)  Xử lý trường hợp số bé trừ số lớn ra kết quả âm:  - Tính ra số bù 1.  - Cộng 1 ra số bù 2.  Phép nhân 2 số.  Phép chia 2 số.  Hàm *IN* in ra giá trị output.  In ra dấu cho trường hợp số âm.  Hàm *Xoa* – xóa bằng cách lặp lại bước in các khoảng trắng.  Sử dụng UART để truyền tín hiệu xuất cho màn hình.  Gía trị cần xuất lưu trong thanh ghi A.  Truyền giá trị cần xuất vào SBUF.  Định nghĩa bảng ký tự:  - #!ROW0  - #!ROW1  - #!ROW2  - #!ROW3 |

1. ***Kết quả thực thi:***

Video mô phỏng lại quá trình mạch chạy trên phần mềm Proteus được đính kèm trong link:

<https://drive.google.com/file/d/1jY5cB-IbMcfIHCYD7upLdfdwcXFNXQyN/view?usp=sharing> A screenshot of a computer

Description automatically generated