Đo lường và tự động điều khiển

BÁO CÁO BÀI TẬP LỚN

Đo và điều khiển tốc độ động cơ dùng vi điều khiển 8051

I.Giới thiệu chung:

1.Mở đầu:

Ngày nay trong mọi lĩnh vực khoa học kỹ thuật luôn xuất hiện khái niệm Kỹ thuật số vi xử lý và điều khiển, với sự trợ giúp của máy tính kỹ thuật vi xử lý và điều khiển đã có sự phát triển ạnh mẽ đặc biệt là sự phát triển nhanh chóng của các họ vi xử lý và điều khiển với những tính năng mới. Để phục vụ tốt cho môn học "**Đo lường và điều khiển tự động"** chúng em thực hiện đề tài: Đo và Điều khiển Tốc Độ Động Cơ với mục đích tích luỹ kiến thức đặc biệt là những kinh nghiệm trong quá trình lắp mạch thực tế song do thời gian và kiến thức có hạn, nên mạch thiết kế còn nhiều thiếu sót. Chúng em rất mong nhận được sự góp ý của các thầy cô để có thể nâng cao chất lượ ng của bài thiết kế, chúng em xin chân thành cảm ơn !

2. Đề tài : Đo và điều khiển tốc độ động cơ một chiều loại nhỏ

3. Nhóm sinh viên thực hiện:

Nhóm thực hiện: Gồm 3 thành viên chính được phân công công việc cụ thể

Đo lường và tự động điều khiển

4.Định hướng thiết kế:

Thiết kế một h ệ vi xử lý bao gồm cả việc thi ết kế tổ ch ức phần cứng và viết phần mềm cho nền ph ần cứng mà ta thiết kế. Việc xem xét giữa tổ chức phần cứng và chương trình phần mềm cho một thiết kế là mộ t vấn đề cần phải cân nhắc. Vì khi tổ chức phần cứng càng phức tạp, càng có nhiều chức năng hỗ tr ợ cho yêu cầu thiết kế thì ph ần mềm càng được giảm bớt và d ễ dàng thực hiện nhưng lại đẩy cao giá thành chi phí cho phần cứng, cũ ng như chi phí bảo trì. Ngược lại với một phần cứng tối thiểu lại yêu cầu mộ t chương trình phần mềm phức tạp hơn, hoàn thiện hơn; nhưng lại cho phép bảo trì hệ th ống dễ dàng hơn cũng như việc phát triển tính năng của hệ thống từ đó có thể đưa ra giá cạnh tranh được.

Từ yêu cầu và nhận định trên ta có những định hướng sơ bộ cho thiết kế như sau:

1. Chọn bộ vi xử lý.

Từ yêu cầu dùng VXL 8 bit ta dự kiến dùng các chip vi đ iều khi ển thuộc h ọ MCS-51 của Intel, mà cụ thể ở đây là dùng chip 8051 vì những lý do sau:

+ Thứ nhất 8051 thuộc họ MCS-51, là chip vi điều khiển. Đặc điểm của các chip vi điều khiển nói chung là nó được tích hợp với đầy đủ chức năng của một hệ VXL nhỏ, rất thích hợp với những thiết kế hướng điều khiển. Tức là trong nó bao gồm: mạch VXL, bộ nhớ chương trình và dữ liệu, bộ đếm, bộ tạo xung, các cổng vào/ra nối tiếp và song song, mạch điều khiển ngắt…

+ Thứ hai là, vi điều khiển 8051 cùng với các họ vi điều khiển khác nói chung trong những năm gần đây được phát triển theo các hướng sau:

- Giảm nhỏ dòng tiêu thụ.
- Tăng tốc độ làm việc hay tần số xung nhịp của CPU

.

- Giảm điệp áp nguồn nuôi.
- Có thể mở rộng nhiều chức năng trên chip, mở rộng cho các thiết kế lớn.

Những đặc đ iểm đó dẫn đến đạt được hai tính năng quan trọng là: giảm công suất tiêu thụ và cho phép điều khiển thời gian thực nên về mặt ứng dụng nó rất thích hợp với các thiết kế hướng điều khiển.

Đo lường và tự động điều khiển

+ Thứ ba là, vi điều khiển thuộc họ MCS-51 được hỗ trợ một tập lệnh phong phú nên cho phép nhiều khả năng mềm dẻo trong vấn đề viết chương trình phần mềm điều khiển.

+ Cuối cùng là, các chip thuộc họ MCS-51 hiện được sử dụng phổ biến và được coi là chuẩn công nghiệp cho các thiết kế khả dụng. Mặt khác, qua việc khảo sát thị trường linh kiện việc có được chip 8051 là dễ dàng nên mở ra khả năng thiết kế thực tế.

Vì những lý do trên mà việc l ựa chọn vi điều khiển 8051 là một giải pháp hoàn toàn phù hợp cho thiết kế.

4 .Phương án thực hiện :

4.1. Dùng cặp cảm bi ến thu phát đặt đối diện để xác định số vòng quay trong một khoảng thời gian nhất định . Động cơ có gắn mộ t đĩa quay có một khe thủ ng trên đĩa ,mỗI khi khe này quay qua cặp cảm biến hồng ngoạt thu phát sẽ tạo ra một đột biến xung trong một vòng quay.

4.2. Sử dụng cảm biến phát và đồng thời thu tín hiệu phản xạ ngược trở bằng cách vạch một số điểm trên trục của động cơ .

4.3. Họ vi điều khiển AT89C51 có 32 đường xuất nhập dữ liệu : P0 ,P1 , P2, P3 mỗI Port 8 bit vì vậy ph ương án đặt ra sử dụng toàn bộ 8 bit P*.0 - P* .7 để xuất ra LED 7 thanh CA hoặc ch ỉ sử d ụng mỗ I Port 4bit sau đó giảI mã bằng 74LS47.Như vậy sẽ phảI sử dụng LCD để hiển thị tốc độ động cơ .

4.4. Sử dụng màn hình LCD để hiển thị .

5. Các bước thực hiện :

Sau khi nhận đồ án nhóm em đã đưa ra một số bước sau để thực hiện công việc:

1.Nhập số vào LCD theo đúng trình tự hàng trăm hàng chục hàng đơn vị .Đo tốc độ của các động cơ loại nh ỏ (loại một chiều hoặc xoay chiều),có gắn cánh quạt (số lượng cánh là xác định).

2. Thực hiện việc đo tốc độ thông qua số vòng quay của cánh quạt bằng cách sử dụng mạch sensor thu phát hồng ngoại.

3.Việc hiển thị thực hiện thông qua LCD (đo tốc độ trong một khoảng thời gian phù hợp).Có một khoảng thời gian để quan sát giá trị của tốc độ.

4.Việc đo động cơ ta điều chỉnh sao cho tốc độ của động cơ luôn ổn định ở một ngưỡng nhất định .Nghĩa là tốc độ của động cơ luôn có một sai số trong giới hạn .trong bài này chúng em điều chỉnh cho sai số của động cơ trong khoảng 2%.

Đo lường và tự động điều khiển

6.Mô phỏng

a.Phần code

```
// Mo PhongDlg.cpp : implementation file
//

#include "stdafx.h"
#include "Mo Phong.h"
#include "Mo PhongDlg.h"
#include"math.h"
#include "stdlib.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
	CAboutDlg();

// Dialog Data
	//{{AFX_DATA(CAboutDlg)
	enum { IDD = IDD_ABOUTBOX };
	//}}AFX_DATA

	// ClassWizard generated virtual function overrides
	//{{AFX_VIRTUAL(CAboutDlg)
	protected:
	virtual void DoDataExchange(CDataExchange* pDX); // DDX/DDV
support
	//}}AFX_VIRTUAL

// Implementation
protected:
	//{{AFX_MSG(CAboutDlg)
```

Đo lường và tự động điều khiển

```
	//}}AFX_MSG
	DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
	//{{AFX_DATA_INIT(CAboutDlg)
	//}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CAboutDlg)
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
	//{{AFX_MSG_MAP(CAboutDlg)
		// No message handlers
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMoPhongDlg dialog

CMoPhongDlg::CMoPhongDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CMoPhongDlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CMoPhongDlg)
		// NOTE: the ClassWizard will add member initialization
	here //}}AFX_DATA_INIT
	// Note that LoadIcon does not require a subsequent DestroyIcon in
Win32
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CMoPhongDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
```

Đo lường và tự động điều khiển

```
	//{{AFX_DATA_MAP(CMoPhongDlg)
	DDX_Control(pDX, IDC_LED1, m_led1);
	DDX_Control(pDX, IDC_LED10, m_led10);
	DDX_Control(pDX, IDC_LED2, m_led2);
	DDX_Control(pDX, IDC_LED3, m_led3);
	DDX_Control(pDX, IDC_LED4, m_led4);
	DDX_Control(pDX, IDC_LED5, m_led5);
	DDX_Control(pDX, IDC_LED6, m_led6);
	DDX_Control(pDX, IDC_LED7, m_led7);
	DDX_Control(pDX, IDC_LED8, m_led8);
	DDX_Control(pDX, IDC_LED9, m_led9);
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CMoPhongDlg, CDialog)
	//{{AFX_MSG_MAP(CMoPhongDlg)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_BN_CLICKED(IDC_HANGCHUC, OnHangchuc)
	ON_BN_CLICKED(IDC_HANGDONVI, OnHangdonvi)
	ON_BN_CLICKED(IDC_HANGNGHIN, OnHangnghin)
	ON_BN_CLICKED(IDC_HANGTRAM, OnHangtram)
	ON_BN_CLICKED(IDC_HANGVAN, OnHangvan)
	ON_BN_CLICKED(IDC_PAUSE, OnPause)
	ON_BN_CLICKED(IDC_RESET, OnReset)
	ON_BN_CLICKED(IDC_REVERSE, OnReverse)
	ON_WM_TIMER()
	ON_BN_CLICKED(IDC_START, OnStart)
	ON_WM_DESTROY()
	ON_BN_CLICKED(IDC_BUTTON1, OnButton1)
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMoPhongDlg message handlers


BOOL CMoPhongDlg::OnInitDialog()
{
```

Đo lường và tự động điều khiển

```
CDialog::OnInitDialog();

// Add "About..." menu item to system menu.

// IDM_ABOUTBOX must be in the system command range.
ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
ASSERT(IDM_ABOUTBOX < 0xF000);

CMenu* pSysMenu = GetSystemMenu(FALSE);
if (pSysMenu != NULL)
{
        CString strAboutMenu;
        strAboutMenu.LoadString(IDS_ABOUTBOX);
        if (!strAboutMenu.IsEmpty())
        {
                pSysMenu->AppendMenu(MF_SEPARATOR);
                pSysMenu->AppendMenu(MF_STRING,
IDM_ABOUTBOX, strAboutMenu);
        }
}

// Set the icon for this dialog. The framework does this automatically
//  when the application's main window is not a dialog
SetIcon(m_hIcon, TRUE);                 // Set big icon
SetIcon(m_hIcon, FALSE);        // Set small icon

// TODO: Add extra initialization
here m_1=0;
m_2=m_3=m_4=m_5=m_6=m_7=m_8=m_9=m_10=0;
m_led1.SetWindowText("0");
m_led2.SetWindowText("0");
m_led3.SetWindowText("0");
m_led4.SetWindowText("0");
m_led5.SetWindowText("0");
m_led6.SetWindowText("0");
m_led7.SetWindowText("0");
m_led8.SetWindowText("0");
m_led9.SetWindowText("0");
m_led10.SetWindowText("0");
m_degree=0;
```

Đo lường và tự động điều khiển

```
	temp=TRUE;
	t=0;
//	m_vong=0;
	return TRUE; // return TRUE unless you set the focus to a control
}

void CMoPhongDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
	if ((nID & 0xFFF0) == IDM_ABOUTBOX)
	{
		CAboutDlg dlgAbout;
		dlgAbout.DoModal();
	}
	else
	{
		CDialog::OnSysCommand(nID, lParam);
	}
}

// If you add a minimize button to your dialog, you will need the code below
//  to draw the icon. For MFC applications using the document/view model,
//  this is automatically done for you by the framework.

void CMoPhongDlg::OnPaint()
{
	if (IsIconic())
	{
		CPaintDC dc(this); // device context for painting

		SendMessage(WM_ICONERASEBKGND, (WPARAM)
dc.GetSafeHdc(), 0);

		// Center icon in client rectangle
		int cxIcon = GetSystemMetrics(SM_CXICON);
		int cyIcon = GetSystemMetrics(SM_CYICON);
		CRect rect;
		GetClientRect(&rect);
		int x = (rect.Width() - cxIcon + 1) / 2;
		int y = (rect.Height() - cyIcon + 1) / 2;
```

Đo lường và tự động điều khiển

```
		// Draw the icon
		dc.DrawIcon(x, y, m_hIcon);
	}
	else
	{
		CClientDC pDC(this);
		OnDraw(&pDC);
		CDialog::OnPaint();
	}
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CMoPhongDlg::OnQueryDragIcon()
{
	return (HCURSOR) m_hIcon;
}

void CMoPhongDlg::OnHangchuc()
{
if(m_4==9) m_4=0;
else
 m_4++;
CString s;
s.Format("%d",m_4);
m_led4.SetWindowText(s);
	m_vong=m_1*10000+m_2*1000+m_3*100+m_4*10+m_5;

}

void CMoPhongDlg::OnHangdonvi()
{
if(m_5==9) m_5=0;
else
 m_5++;
CString s;
s.Format("%d",m_5);
m_led5.SetWindowText(s);
	m_vong=m_1*10000+m_2*1000+m_3*100+m_4*10+m_5;
```

```
}

void CMoPhongDlg::OnHangnghin()
{
if(m_2==9) m_2=0;
else
 m_2++;
CString s;
s.Format("%d",m_2);
m_led2.SetWindowText(s);
        m_vong=m_1*10000+m_2*1000+m_3*100+m_4*10+m_5;

}

void CMoPhongDlg::OnHangtram()
{
if(m_3==9) m_3=0;
else
 m_3++;
CString s;
s.Format("%d",m_3);
m_led3.SetWindowText(s);
m_vong=m_1*10000+m_2*1000+m_3*100+m_4*10+m_5;

}


void CMoPhongDlg::OnHangvan()
{
        if(m_1==9) m_1=0;
else
 m_1++;
CString s;
s.Format("%d",m_1);
m_led1.SetWindowText(s);
        m_vong=m_1*10000+m_2*1000+m_3*100+m_4*10+m_5;

}

void CMoPhongDlg::OnPause()
```

```
{
	KillTimer(1);
}

void CMoPhongDlg::OnReset()
{
	m_1=m_2=m_3=m_4=m_5=m_6=m_7=m_8=m_9=m_10=0;
	// thiet lap ve khong
	m_led1.SetWindowText("0");
	m_led2.SetWindowText("0");
	m_led3.SetWindowText("0");
	m_led4.SetWindowText("0");
	m_led5.SetWindowText("0");
	m_led6.SetWindowText("0");
	m_led7.SetWindowText("0");
	m_led8.SetWindowText("0");
	m_led9.SetWindowText("0");
	m_led10.SetWindowText("0");
	KillTimer(1);
}

void CMoPhongDlg::OnReverse()
{
	if(temp==TRUE)
		temp=FALSE;
	else
	{
		temp=TRUE;
	}
}

void CMoPhongDlg::OnDraw(CDC *pDC)
{
	CRect rectWin;
	GetWindowRect(rectWin);
	rectWin.top+=32;
	CWnd*pWnd=GetDlgItem(IDC_DISPLAY);
	CRect rectD;
	pWnd->GetWindowRect(rectD);
	int cx,cy;
```

Đo lường và tự động điều khiển

```
        cx=rectD.CenterPoint().x;
        cy=rectD.CenterPoint().y;
        cx-=rectWin.left;
        cy-=rectWin.top;
        CBrush brush(RGB(0,255,0));
        CBrush * oldBrush=pDC->SelectObject(&brush);
        CPen *pen=new CPen(PS_SOLID,1,RGB(0,0,255));
        CPen*oldPen =pDC->SelectObject(pen); pDC-
        >Ellipse(cx -50,cy-50,cx+50,cy+50);
        CPen*pen1=new CPen(PS_SOLID,3,RGB(255,0,0));
        CPen*oldPen1 =pDC->SelectObject(pen1);

        double angle;
        angle =m_degree*3*3.14/180;
        pDC->MoveTo(cx,cy);
        pDC->LineTo(cx+(int)(50*sin(angle)),cy-(int)(50*cos(angle)));
        angle+=3.14/3;
        pDC->MoveTo(cx,cy);
        pDC->LineTo(cx+(int)(50*sin(angle)),cy-(int)(50*cos(angle)));
        angle+=3.14/3;
        pDC->MoveTo(cx,cy);
        pDC->LineTo(cx+(int)(50*sin(angle)),cy-(int)(50*cos(angle)));
        angle+=3.14/3;
        pDC->MoveTo(cx,cy);
        pDC->LineTo(cx+(int)(50*sin(angle)),cy-(int)(50*cos(angle)));
        angle+=3.14/3;
                pDC->MoveTo(cx,cy);
        pDC->LineTo(cx+(int)(50*sin(angle)),cy-(int)(50*cos(angle)));
        angle+=3.14/3;
        pDC->MoveTo(cx,cy);
        pDC->LineTo(cx+(int)(50*sin(angle)),cy-(int)(50*cos(angle)));
        pDC->MoveTo(cx,cy);
        pDC->LineTo(cx+(int)(50*sin(angle)),cy-(int)(50*cos(angle)));
        pDC->SelectObject(oldBrush);
        pDC->SelectObject(oldPen);
        pDC->SelectObject(oldPen1);

}

void CMoPhongDlg::OnTimer(UINT nIDEvent)
```

Đo lường và tự động điều khiển

```
{
        if(temp==TRUE)
        {
        m_degree++;
        if(m_degree>=120) m_degree=0;
        }
        else
        {
                m_degree--;
                if(m_degree<=0) m_degree=120;
        }
        if(m_10<9)
                m_10++;
        else
        {
                m_10=0;
                if(m_9<9)
                m_9++;
                else
                {
                        m_9=0;
                        if(m_8<9)
                                m_8++;
                        else
                        {
                                m_8=0;
                                if(m_7<9)
                                        m_7++;
                                else
                                {
                                        m_7=0;
                                        if(m_6<9)
                                        m_6++;
                                        else
                                                MessageBox("Tran
So","Warning",MB_OK);
                                }
                        }
                }
        }
```

Đo lường và tự động điều khiển

```
        t++;

        if(t==100)
        {
                t=0;
        CString s6,s7,s8,s9,s10;
        s6.Format("%d",m_1);
        s7.Format("%d",m_2);
        s8.Format("%d",m_3);
        s9.Format("%d",m_4);
        if(m_5<=2)
                        s10.Format("%d",m_5+1);
        else
        if((m_5>2)&&(m_5<5))
                s10.Format("%d",m_5-1);
        else
                if(m_5==5||m_5==7)
                        s10.Format("%d",m_5);
                else
                if((m_5>5)&&m_5<7)
                        s10.Format("%d",m_5+1);
        else
                s10.Format("%d",m_5-1);

        m_led6.SetWindowText(s6);
        m_led7.SetWindowText(s7);
        m_led8.SetWindowText(s8);
        m_led9.SetWindowText(s9);
        m_led10.SetWindowText(s10);
        }
        CClientDC pDC(this);
        OnDraw(&pDC);
        CDialog::OnTimer(nIDEvent);

}

void CMoPhongDlg::OnStart()
{
        UpdateData(TRUE);
        SetTimer(1,(int)(1000/m_vong),NULL);
```

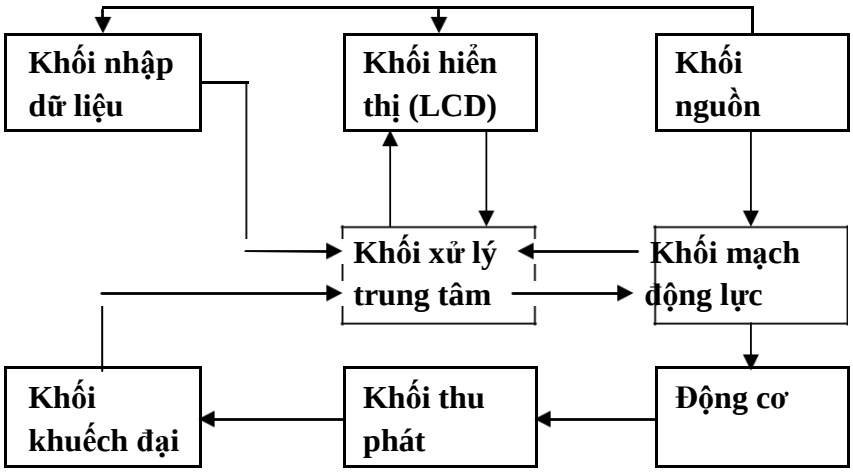
```
}

CScrollBar* CMoPhongDlg::GetScrollBarCtrl(int nBar) const
{
	// TODO: Add your specialized code here and/or call the base class
	return CDialog::GetScrollBarCtrl(nBar);
}

void CMoPhongDlg::OnDestroy()
{
	CDialog::OnDestroy();

	KillTimer(1);
}

void CMoPhongDlg::OnButton1()
{
OnOK();
}
```

b. Phần giao diện

II. Lý Thuyết thực hiện.

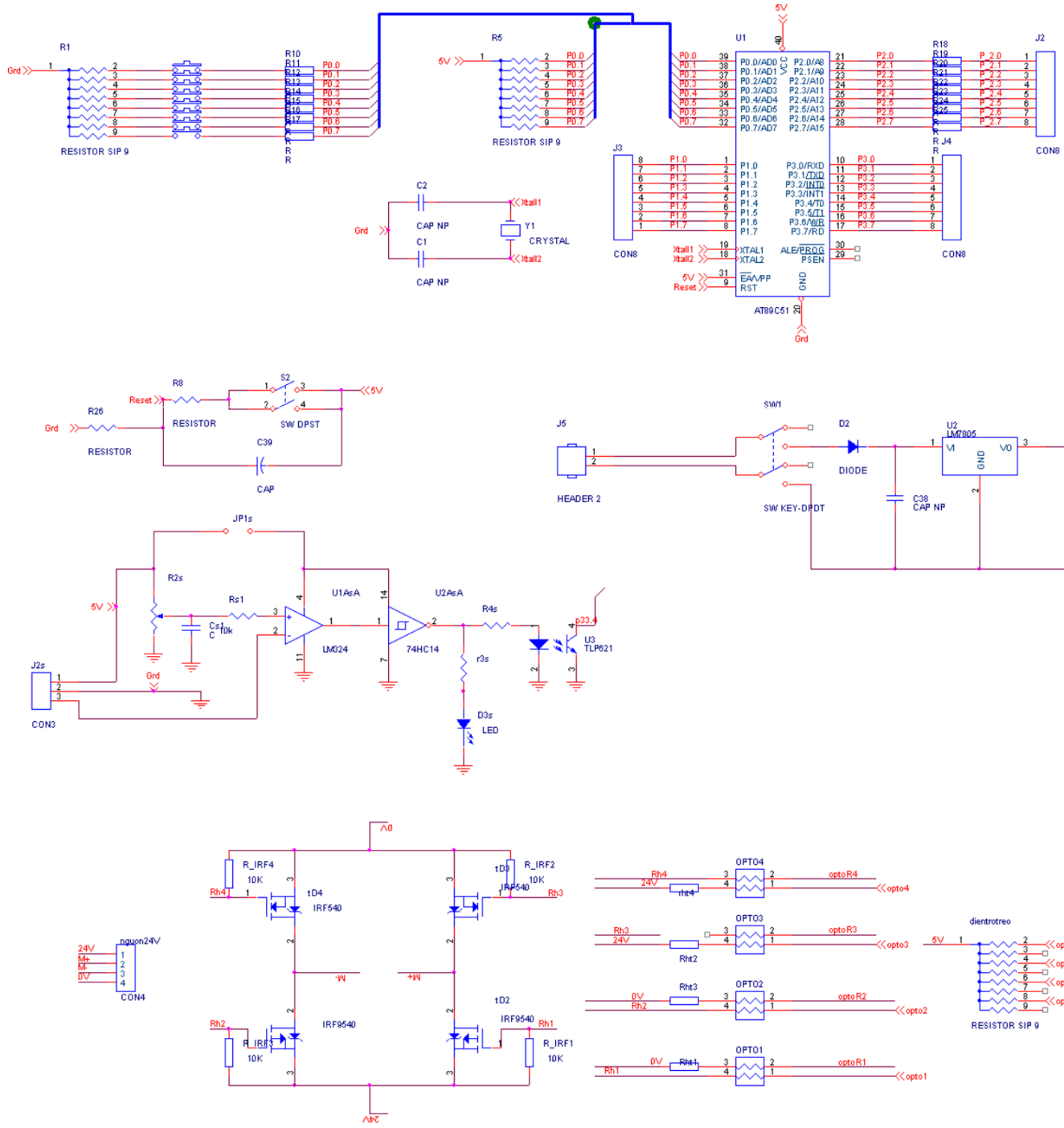
1. Cơ sở lý thuyết. Sơ đồ khối

Đo lường và tự động điều khiển



Đo lường và tự động điều khiển

Sơ đồ nguyên lý.



Đo lường và tự động điều khiển

III.Phân tích chức năng từng khối.

a .Khối vi xử lý
b. Khối hiển thị.
c.Khối nhập giá trị tốc độ.
d.Khối đo tốc độ động cơ.
e.Khối động lực
g.Khối nguồn
h.Khối động cơ

IV.Giới thiệu linh kiện sử dụng trong mạch:

1 - IC khuếch đại LM324
2 - Vi điều khiển 80c52
3 – LCD. ,74HC14,điện trở quang….
4 - Một số linh kiện phụ khc: thạch anh 12 Mhz, sensor thu phát hồng ngoại, tụ 33p, tụ 10 uF, trở...
5 - Môtơ điện một chiều DC 12V

V .Mô tả các modul

1.Khối vi xử lý Điều khiển tốc độ động cơ:

1. Giới thiệu:

Mục đích của điều khiển tốc độ động cơ là đưa ra tín hiệu biểu diễn tốc độ yêu cầu, và đi ều khiển động cơ theo tốc độ đấy. Bộ điều khiển có thể có hoặc không th ật sự đo tốc độ độ ng cơ. Nếu có thì goi là điều khiển t ốc độ có phản hồi hoặc điều khiển tốc độ vòng kín, nếu không thì gọi là điều khiển t ốc độ vòng mở. Điều khiển tốc độ có phản hồi tốt hơn nhưng phức tạp hơn. Động cơ có rất nhiều kiểu, và đầu ra của bộ điều khiển tốc độ của động cơ với các d ạng khác nhau là khác nhau.

2. Lý thuyết điều khiển tốc độ động cơ một chiều:

Tốc độ củ a động cơ một chiều tỉ lệ trực ti ếp với nguồ n cấp, vì vậy nếu ta giảm điện áp cung cấp từ 12V xuống 6V, động cơ sẽ chạy với tốc độ bằng một nửa trước đó.

Bộ đ iều khiển tốc độ động cơ làm việc trên nguyên lý biến đổi đ iện áp trung bình cấp cho độ ng cơ. Có thể đơn giản chỉ bằng cách đ iều chỉnh điện áp cung cấp, nhưng như thế sẽ không hiệu quả. Cách tốt hơn là tắt nguồn cấp cho

động cơ thật nhanh. Nếu động tác tắt này đủ nhanh thì động cơ không kịp nhận ra sự thay đổi đó mà chỉ nhận ra được hi ệu ứng trung bình thôi. Khi bật, nguồn có giá trị 12V; khi tắt, nguồn có giá tr ị 0V. Nếu ta tắt nguồn với một lượng thời gian b ằng với khi nó được bật thì động cơ sẽ nhận được giá trị trung bình là 6V, và sẽ chạy chậm đi theo tỉ lệ đó.
Chuyển mạch để bật tắt nguồn này gọi là on-off switching, được chế tạo bằng MOSFET.

Ta dùng vi xử lý 8051.

Những tính chất đặc trưng của họ vi điều khiển MCS-51:

* Đơn vị xử lý trung tâm (CPU) 8 bit đã được tối ưu hoá để đáp ứng các chức năng điều khiển .

* Khối lôgic (ALU) xử lý theo bit nên thuận tiện cho các phép toán logic Boolean.

* Bộ tạo dao động giữ nhịp được tích hợp bên trong với tần số 12MHz.

* Giao diện nối tiếp có khả năng hoạt động song song, đồng bộ.

* Các cổng vào/ra hai hướng và từng đường dẫn có thể được định địa chỉ một cách tách biệt.

* Có năm hay sáu nguồn ngắt với hai mức ưu tiên .

* Hai hoặc ba bộ đếm định thời 16 bit.

* Bus và khối định thời tương thích với các khối ngoại vi của bộ vi xử lý 8085/8088.

* Dung lượng của bộ nhớ chương trình (ROM) bên ngoài có thể lên tới 64 kbyte.

* Dung lượng của bộ nhớ dữ liệu (RAM) bên ngoài có thể lên tới 64 kbyte.

* Dung lượng của bộ nhớ ROM bên trong có thể lên đến 8 kbyte.

* Dung lượng bộ nhớ RAM bên trong có thể đạt đến 256 byte.

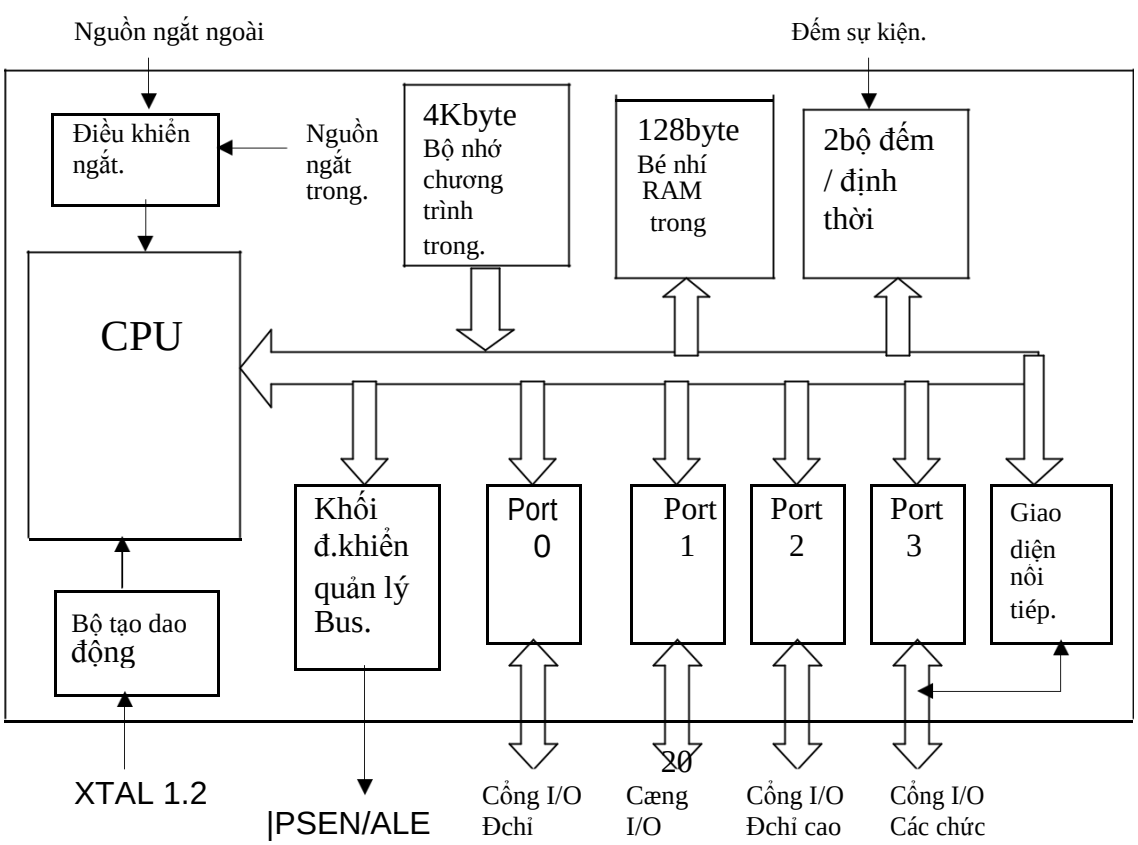
* Tập lệnh phong phú.

2.1. Cấu trúc chung :

Đo lường và tự động điều khiển

2.1.1. Sơ đồ khối :

S ơ đồ khối tổng quát của một vi điều khiển 8051 có thể được mô tả như sau:



thấp	8 bit	Dữ liệu 8	năng đắc biệt

Đo lường và tự động điều khiển

Chức năng của từng khối :

* Khối xử lý trung tâm CPU:

Phần chính của bộ vi xử lý là khối xử lý trung tâm (CPU=Central Processing Unit), khối này có chứa các thành phần chính :

+Thanh ghi tích luỹ (ký hiệu là A);

+Thanh ghi tích luỹ phụ (ký hiệu là B) thường được dùng cho phép nhân và phép chia ;

+Khối logic số học (ALU=Arithmetic Logical Unit) ;

+Từ trạng thái chương trình (PSW= Program Status Word); +Bốn băng thanh ghi .

+Con trỏ ngăn x ếp (SP=Stack Point) cũng như con trỏ dữ liệu để định địa chỉ cho bộ nhớ dữ liệu ở bên ngoài;

Ngoài ra, khối xử lý trung tâm còn chứa:

-Thanh ghi đếm chương trình (PC= Progam Counter);

-Bộ giải mã lệnh;

-Bộ điều khiển thời gian và logic;

Sau khi được Reset, CPU b ắt đầu làm việc tại địa chỉ 0000h, là địa chỉ đầu được ghi trong thanh ghi chứa ch ương trình (PC) và sau đó, thanh ghi này sẽ tăng lên 1 đơn vị và chỉ đến các lệnh tiếp theo của chương trình.

*Bộ tạo dao động:

Đo lường và tự động điều khiển

Khố i xử lý trung tâm nhậ n trực tiếp xung nhịp từ bộ tạo dao động đượ c lắp thêm vào, linh kiện phụ trợ có thể là một khung dao động làm bằng tụ gốm hoặc th ạch anh. Ngoài ra, còn có thể đưa một tín hiệu giữ nhịp từ bên ngoài vào.

*Khối điều khiển ngắt:

Chươ ng trình đang chạy có thể cho dừng lại nhờ một khố i logic ng ắt ở bên trong. Các nguồ n ng ắt có thể là: các biến cố ở bên ngoài, sự tràn bộ đếm/bộ định thời hay có th ể là giao diện n ối ti ếp. Tất cả các ng ắt đều có thể được thiết lập chế độ làm việc thông qua hai thanh ghi IE (Interrupt Enable) và IP (Interrupt Priority).

*Khối điều khiển và quản lý Bus :

Các kh ối trong vi điều khiển liên lạc với nhau thông qua hệ thống Bus nội bộ được điều khiển bởi khối điều khiển quản lý Bus.

*Các bộ đếm/định thời:

Vi đi ều khiển 8051 có chứa hai b ộ đếm ti ến 16 bit có thể hoạt động nh ư là bộ định thời hay bộ đếm sự kiện bên ngoài hoặc như bộ phát tốc độ Baud dùng cho giao diện nối tiếp. Trạng thái tràn bộ đếm có thể được kiểm tra trực tiếp hoặc được xoá đi bằng một ngắt.

*Các cổng vào/ra:

Vi điều khiển 8051 có bốn cổng vào/ra (P0 .. P3), mỗi cổng chứa 8 bit, độc lập với nhau. Các cổng này có thể được sử dụng cho những mục đích điều khiển rất đa dạng. Ngoài chức năng chung, một số cổng còn đảm nhận thêm một số chức năng đặc biệt khác.

*Giao diện nối tiếp:

Giao diện nối tiếp có chứa một bộ truyền và mộ t bộ nhận không đồng bộ làm việc độc lập với nhau. Bằng cách đấu nối các bộ đệm thích h ợp, ta có thể hình thành một cổ ng nối tiếp RS -232 đơn giản. Tốc độ truyền qua cổng nối tiếp có thể đặt được trong một vùng rộng phụ thuộc vào một bộ định thời và tần số dao động riêng của thạch anh.

*Bộ nhớ chương trình:

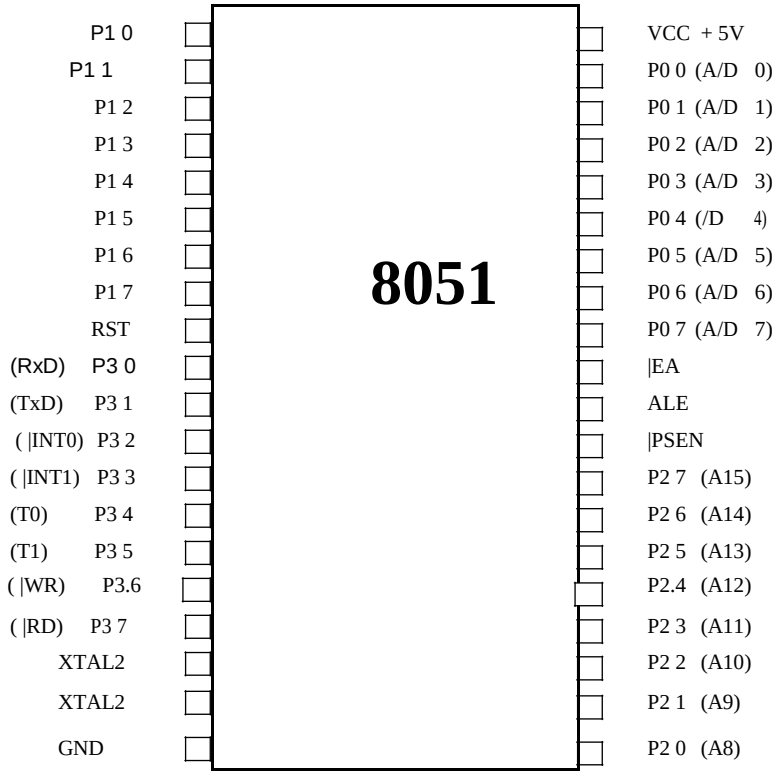
Bộ nhớ chương trình thường là bộ nhớ ROM (Read Only Memory), bộ nhớ chương trình được sử dụng để cất giữ chương trình điều khiển hoạt động của vi điều khiển.

*Bộ nhớ số liệu:

Bộ nhớ số liệu th ường là bộ nhớ RAM (Ramdom Acces Memory), b ộ nh ớ số li ệu dùng để cất giữ các thông tin tạm thời trong quá trình vi điều khiển làm việc.

2.1.2. Sự sắp xếp chân ra của vi điều khiển 8051:

Phần lớn các bộ vi điều khiển 8051 được đóng vào vỏ theo kiểu hai hàng DIL(Dual In Line) với tổng số là 40 chân ra, một số ít còn lại được đóng vỏ theo kiểu hình vuông PLCC (Plastic Leaded Chip Carrier) v ới 44 chân và loại này thường được dùng cho những hệ thống cần thiết phải tiết kiệm diện tích.



Hình 2.2: Sơ đồ chân của vi mạch 8051 DIL.

Bảng 2.1: Chức năng các chân của vi điều khiển 8051.

	Ký hiệu	Chức năng
1-->8	P1.0-->P1.7	Cổng giả hai hướng P1, có thể tự do sử dụng
9	Reset	Lối vào Reset, khi hoạt động ở mức High(1)
10-->17	P3.0-->P3.7	Cổng giả hai hướng P3, sắp xếp tất cả các đường dẫn với chức năng đặc biệt

Đo lường và tự động điều khiển

18	XTAL2	Lối ra của bộ dao động thạch anh bên trong
19	XTAL1	Lối vào của bộ dao động thạch anh bên trong
20	Vss	Nối mát (0V)
21-->28	P2.0-->P2.7	Cổng giả hai hướng P2, chức năng đặc biệt là các đường dẫn địa chỉ A8..A15
29	|PSEN	Progam Strobe Enable, xuất ra các xung đọc dùng cho bộ nhớ chương trình bên ngoài
30	ALE	Address Latch Enable, xuất ra các xung điều khiển để lưu trữ trung gian các địa chỉ
31	|EA	External Access, khi được nối với mát là để làm việc với ROM ngoại vi
32-->39	P1.0-->P1.7	Cổng hai hướng cực máng hở P0 hay Bus dữ liệu hai hướng dùng cho ROM, RAM và thiết bị ngoại vi đồng thời cũng chuyển giao 8 bit địa chỉ thấp
40	Vdd	Nguồn nuôi dương (+5V)

Các chân ra của bộ vi điều khiển 8051 gồm có:

*EA: Đóng vai trò quy ết định xem vi điều khiển làm vi ệc với chương trình bên trong hay bên ngoài. Với loại 8051 không có ROM trong thì chân này phải được nối với mát. Loại thông thường có thể làm việc tuỳ theo cách lựa chọn giữa ROM trong hay ROM ngoài, khi đang ở chế độ làm việc với bộ nhớ ROM trong, loại có chứa bộ nhớ ROM có thể truy nhập tự động lên bộ nhớ chương trình bên ngoài.

*Reset: Trạng thái Reset được thiết lập bằng cách giữ tín hiệu Reset ở mức cao trong thời gian ít nhất là 2 chu kỳ máy.

*ALE: Tín hiệu chốt 8 bit địa chỉ thấp trong suốt quá trìng truy nhập bộ nhớ mở rộng. Thông thường tín hiệu ALE được phát ra với tần số bằng 1/6 tần số dao động thạch anh và có thể sử dụng với mục đích định thời gian hoặc xung nhịp đồng hồ ngoài. Tuy nhiên, tín hiệu ALE sẽ bị bỏ qua trong mỗi quá trình truy nhập bộ nhớ dữ liệu ngoài.

*PSEN: Tín hiệu đọc bộ nhớ chương trình ngoài, khi vi điều khiển truy nhập bộ nhớ chương trình nội thì PSEN được đặt ở mức cao.

*XTAL1, XTAL2: M ột bộ tạo tín hiệu gi ữ nh ịp v ới t ần s ố được xác định bởi bộ cộng hưởng thạch anh được lắp thêm vào, tần số này xác định tốc độ

làm củ a bộ vi đi ều khiển. Thông thường các lệnh được thực hiện bằng 1/12 tần số dao động của thạch anh.

Các bộ đếm cố thể làm việc trong nhiều chế độ khác nhau. Khi hoạt động như là bộ định thời, các bộ đếm nh ận được các xung từ một bộ chia trước ở bên trong, bộ này chia tần số riêng của bộ cộng hưởng thạch anh cho 12 .

Thay cho một bộ định thời 16 bit, một bộ đinh thời 8 bit có thể được tạo ra bằng việc nạp tự động sau khi cấp nguồn, các xung dẫn từ bên ngoài vào qua T0 và T1 cũng có thể được đếm, các xung này có tần số cực đại bằng 1/24 giá trị tần số của bộ cộng hưởng thạch anh.

*P0..P3: Các công vào/ra.

Cổng P3 cũng đảm nhận một số chức năng đặc biệt của bộ vi điều khiển :

Chân	Ký hiệu	Chức năng
P3.0	RxD	Nhận dữ liệu vào bộ nhớ qua cổng nối tiếp
P3.1	TxD	Truyền dữ liệu vào bộ nhớ qua cổng nối tiếp
P3.2	|INT0	Ngắt ngoài 0
P3.3	|INT1	Ngắt ngoài 1
P3.4	T0	Lối vào của Timer 0
P3.5	T1	Lối vào của Timer 1
P3.6	|WR	Viết vào bộ nhớ
P3.7	|RD	Đọc bộ nhớ

2.2 Tổ chức bộ nhớ:

2.2.1. Cấu trúc chung của bộ nhớ:

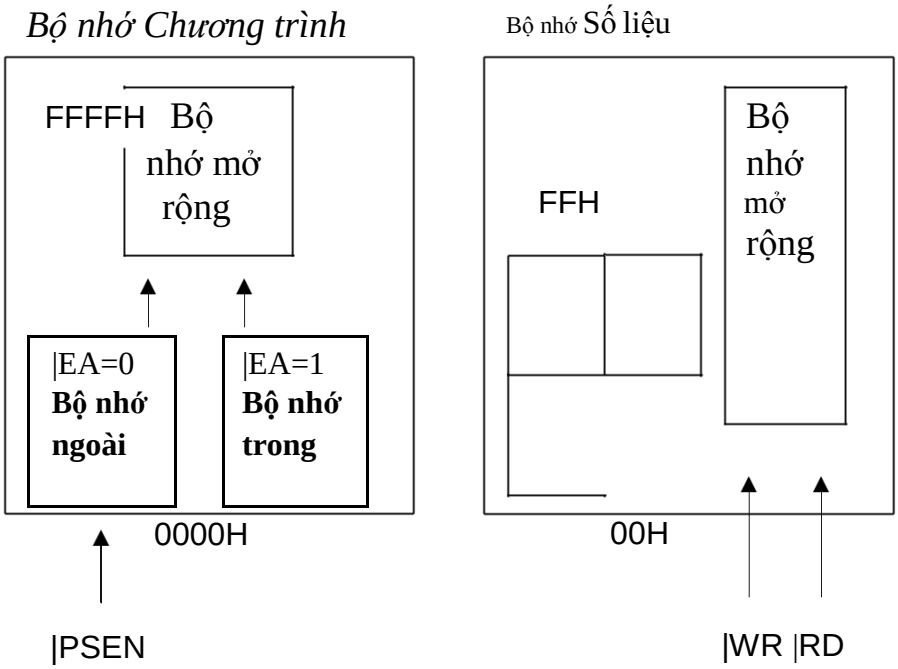
Tất cả các vi đ iều khiển thuộc họ MCS-51 đều phân chia bộ nhớ thành hai vùng địa chỉ cho bộ nh ớ d ữ liệu và bộ nhớ chương trình. Sự phân chia logic giữa bộ nh ớ d ữ li ệu và bộ nhớ chương trình cho phép truy nhập bộ nhớ dữ liệu bằng 8 bit địa chỉ giúp cho việc lưu trữ và thao tác dữ liệ u nhanh hơn.Tuy nhiên, chúng ta có thể sử dụng địa chỉ bộ nhớ dữ liệu 16 bit thông qua thanh ghi DPTR.

Bộ nhớ chươ ng trình là loại bộ nhớ chỉ cho phép đọc, không cho phép ghi. Một số vi đ iều khiển được tích hợp sẵn bộ nhớ ch ương trình bên trong với dung lượng khoảng 4kbyte hay 8 kbyte, số còn lại phải sử dụng bộ chương

trình mở rộng mà quá trình truy nhập được thực hiện thông qua sự điều khiển bằng tín hiệu PSEN (Progam Strobe Enable).

Tuy nhiên, vi đi ều khiển 8051 cho phép ta sử dụ ng đến 64kbyte bộ nhớ chương trình bằng cách sử dụng cả bộ nhớ chương trình bên trong và bên ngoài.

Bộ nhớ số liệu chiếm giữ vùng địa chỉ phân chia của bộ nhớ ch ương trình. Dung lượ ng củ a b ộ nhớ dữ liệu có thể mở rộng lên tới 64 kbyte. Trong quá trình truy nhập bộ nhớ s ố liệu, CPU phát ra các tín hiệu đọc và tín hiệu viết số liệu thông qua các chân RD và WR.



Hình 2.3: *Cấu trúc bộ nhớ của họ MCS-51.*

Chúng ta có thể kết hợp bộ nhớ chương trình mở rộng với bộ nhớ số liệu mở rộng bằng cách cho hai tín hiệu RD và PSEN qua mộ t cổng logic AND, lối ra của cổng AND này sẽ tạo tín hiệu đọc cho bộ nhớ mở rộng.

Đo lường và tự động điều khiển

2.2.2. Bộ nhớ chương trình:

Sau khi Reset, CPU bắt đầu thực hiện chương trình từ địa chỉ 0000H. Vùng đầu của bộ nh ớ chương trình là vùng chứa các vector ngắt, mỗ i ngắt đượ c phân chia mộ t vùng địa ch ỉ cố định trong trong bộ nhớ chương trình. Khi xuất hiện ngắt, CPU sẽ nhảy tới địa ch ỉ này, đây cũng là địa chỉ đầu của chương trình con phục vụ ngắt. Các vector ngắt cách nhau 8 byte, vì vậy nếu chương trình con phục vụ ngắt quá dài (>8 byte) thì tại vector ngắt ta phải đặt một l ệnh nhảy không điều kiện tới vùng địa chỉ khác chứa chương trình con phục vụ ngắt.

2.2.3. Bộ nhớ số liệu:

Phía bên phải củ a *Hình 2.3* biểu di ễn không gian bộ nhớ dữ li ệu của MCS-51. Chúng ta có thể sử d ụng tới 64 Kbyte bộ nhớ số liệu ngoại vi. Độ rộng bus địa chỉ của bộ nh ớ s ố liệu ngoài có thể là 8 bit hoặc 16 bit. Bus địa ch ỉ rộng 8 bit th ường được s ử dụng để liên kết với một hoặc nhiều đường vào ra khác để định địa chỉ cho RAM theo trang. Trong trườ ng hợp bus đị a chỉ rộng 16 bit, cổng P2 sẽ phát ra 8 bit địa ch ỉ cao còn cổng P1 sẽ phát ra 8 bit địa chỉ th ấp. Bằng cách này, ta có thể truy nhập trực tiếp lên bộ nhớ dữ liệu ngoài với độ lớn tối đa là 64 Kbyte.

Bộ nhớ số liệu trong được chia ra làm 3 vùng:

+128 byte cao.

+128 byte thấp.

+Vùng dành cho các thanh ghi chức năng đặc biệt (SFR).

Địa chỉ củ a bộ nhớ số liệu trong luôn là 8 bit, và có thể quản lý được 256 byte bộ nhớ.

Tuy nhiên, trên thực t ế cách định địa chỉ của bộ nhớ RAM trong có thể quản lý tới 384 byte.

Bản đồ bộ nhớ trên chíp:

Đo lường và tự động điều khiển

7F										FF										
										F0	F7	F6	F5	F4	F3	F2	F1	F0	B	
	RAM đa dụng																			
										E0	E7	E6	E5	E4	E3	E2	E1	E0	ACC	
											D0	D7	D6	D5	D4	D3	D2	D1	D0	PSW
30										B8	-	-	-	BC	BB	BA	B9	B8	IP	
2F	7F	7E	7D	7C	7B	7A	79	78												
2E	77	76	75	74	73	72	71	70		B0	B7	B6	B5	B4	B3	B2	B1	B0	P.3	
2D	6F	6E	6D	6C	6B	6A	69	68												
2C	67	66	65	64	63	62	61	60		A8	AF			AC	AB	AA	A9	A8	IE	
2B	5F	5E	5D	5C	5B	5A	59	58												
2A	57	56	55	54	53	52	51	50		A0	A7	A6	A5	A4	A3	A2	A1	A0	P2	
29	4	4	4	4	4	4	4	48												

Đo lường và tự động điều khiển

	F	E	D	C	B	A	9											
28	47	46	45	44	43	42	41	40		99	Không được địa chỉ hoá bit							SBUF
27	3F	3E	3D	3C	3B	3A	39	38		98	9F	9E	9D	9C	9B	9A	99	98	SCON
26	37	36	35	34	33	32	31	30										
25	2F	2E	2D	2C	2B	2A	29	28		90	97	96	95	94	93	92	91	90	P1
24	27	26	25	24	23	22	21	20										
23	1F	1E	1D	1C	1B	1A	19	18		8D	Không được địa chỉ hoá bit							TH1
22	17	16	15	14	13	12	11	10		8C	Không được địa chỉ hoá bit							TH0
21	0F	0E	0D	0C	0B	0A	09	08		8B	Không được địa chỉ hoá bit							TL1
20	07	06	05	04	03	02	01	00		8A	Không được địa chỉ hoá bit							TL0
1F	Bank 3									89	Không được địa chỉ hoá bit							TMOD
18										88	8F	8E	8D	8C	8B	8A	89	88	TCON
17	Bank 2									87	Không được địa chỉ hoá bit							PCON
10																		
0F	Bank 1									83	Không được địa chỉ hoá bit							DPH

08			82	Không được địa chỉ hoá bit	DPL
07	Bank thanh ghi 0		81	Không được địa chỉ hoá bit	SP
00	(Mặc định cho R0 -R7)		88	80	P0

2.2.4. Thanh ghi ghi chức năng đặc biệt (SFR= Special Function Registers):

Vi đ iều khiển 8051 là một bộ vi điều khiển đa năng với nhiều chế độ hoạt động khác nhau được thiết lập thông qua các thanh ghi chức năng đặc biệt SFRs.

Các thanh ghi chức năng đặc biệt của vi điều khiển 8051 gồm có:

+P0, P1, P2, P3: Các cổng vào ra, mỗi bít ứng với 1 chân của vi điều khiển. Các chân này hoạt động ở mức logic âm.

+SP (Stack Pointer): Đây là con trỏ ngăn xếp của vi điều khiển. Thanh ghi này cho biết địa chỉ truy nhập tiếp theo trong bộ nhớ RAM.

+DPH, DPL (Data Pointer High, Data Pointer Low): Tạo thành 1 cặp thanh ghi con trỏ dữ liệu DPTR 16 bit.

+PCON (Power Control): Thanh ghi này được sử dụng để điều khiển công suất của vi điều khiển. Ngoài ra bit PCON.7 con cho phép sủ dụng để tăng gấp đôi tốc độ baud khi truyền qua cổng nối tiếp.

+TCON (Timer Control): Thiết lập cấu hình làm việc cho bộ Timer/Counter.

+TMOD (Timer Mode): Xác định chế độ làm việc cho từng bộ Timer/Counter.

+TL0/TH0 (Timer 0 Low/High): Cặp thanh ghi tương ứng với Timer0.

+TL1/TH1 (Timer 1 Low/High): Cặp thanh ghi tương ứng với Timer1.

+SCON (Serial Control): Thiết lập cấu hình cho cổng nối tiếp.

+SBUF (Serial Buffer): Bộ đệm khi truyền hoặc nhận dữ liệu qua cổng nối tiếp.

+IE (Interrupt Enable): Cho phép ngắt hoặc cấm ngắt.

+IP (Interrupt Priority): Thiết lập mức ưu tiên cho các ngắt.

+PSW (Program Status Word): Thanh ghi từ trạng thái chương trình lưu tr ữ một s ố bit quan trọng được đặt hoặc xoá bởi các l ệnh của 8051: cờ nhớ, cờ nh ớ ph ụ, cờ tràn và cờ chẵn lẻ. Ngoài ra, 2 bit RS0 và RS1 trong PSW còn cho phép chọn băng thanh ghi để làm việc trong bộ nhớ RAM trong.

+ACC (Accumulator): Thanh ghi tích luỹ, đây là một trong những thanh ghi được sử d ụng nhiều nhất trong vi điều khiển 8051. Thanh ghi này có ký hiệu là A.

+B (B Register): Thanh ghi B được sử dụng khi thực hiện các phép toán nhân, chia và cũng có thể được dùng như thanh ghi phụ hay thanh ghi lưu trữ số liệu tạm thời.

Các thanh ghi chức năng đặc biệt sẽ có thể đượ c nhận mộ t trạng thái nào đó cố định mỗi khi vi điều khi ển Reset (tuỳ thuộc vào mỗi thanh ghi). Sau đây là bảng trạng thái của các thanh ghi ngay sau khi Reset:

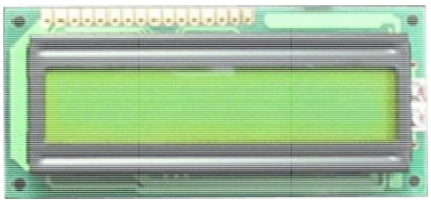
Bảng 2.2: Trạng thái khi reset của các thanh ghi.

Register	**Reset to**	**Register**	**Reset to**	**Register**	**Reset to**
A	00H	P2	FFH	SCON	00H
B	00H	P3	FFH	TCON	00H
DPTR	00H	PCON	0.....B	TMOD	00H
IE	0..00000B		0...0000B	TH0	00H
IP	...00000B	PSW	00H	TH1	00H
P0	FFH	SP	07H	TL0	00H
P1	FFH	SBUF	..H	TH1	00H

2.Khối hiển thị

Đo lường và tự động điều khiển

N ối ghép LCD với vi xử lý 8051: LCD gồm có hai hàng hàng đơn v ị, hàng ch ục, hàng trăm. Hàng đầu là giá tr ị nhập vào để động c ơ chạy, hàng thứ hai là giá trị m à thực chấ t động cơ chạy và đ ược hiển t hị trên LC D.



1. Hoạ t động củ a LCD:

Trong nhữ ng năm g ần đây, mà n hình tin h thể lỏng LCD (Liquid Crystal Display) n gày càng được sử d ụ ng rộng r ãi và đang dần thay thế các đèn LED (7 thanh và nhiều th anh). Đó l à do các n guyên nhâ n sau: -Màn hình LCD có giá thành h ạ
-Khả năng hiển thị số, ký tự và đồ hoạ tốt h ơn nhiều so với đ èn LED (đèn LED chỉ h iển thị đư ợc số và m ột số ký t ự).
-Sử dụng thêm một bộ điều khiển làm tươ i LCD và như vậy giải phóng CPU khỏ i công việc này. Còn đối v ớ i đèn LE D luôn cần CPU (hoặ c bằng cá ch nào đó) để du y trì vieec hiển thị dữ liệu.
-Dễ dàng lập trình c ác ký tự và đồ hoạ.

2. Mô tả chân củ a LCD:

LCD giới thiệu ở đây có 14 chân. Chức năng của các chân đư ợc trình bày trong bản g như sau:

Chân	Ký hiệu	I/O	Mô tả
1	V SS	-	Đất
2	V CC	-	Dương nguồn +5V
3	V EE	-	Nguồn đ iều khiển tương phản
4	RS	I	RS = 0 chọn thanh ghi lệnh RS = 1 chọn thanh ghi dữ liệ u
5	R /W	I	R/W = 1 đọc dữ liệu R/W = 0 ghi
6	E	I/O	Cho phé p
7	D B0	I/O	Bus dữ liệu 8 bits
8	D B1	I/O	Bus dữ liệu 8 bits
9	D B2	I/O	Bus dữ liệu 8 bits

10	DB3	I/O	Bus dữ liệu 8 bits
11	DB4	I/O	Bus dữ liệu 8 bits
12	DB5	I/O	Bus dữ liệu 8 bits
13	DB6	I/O	Bus dữ liệu 8 bits
14	DB7	I/O	Bus dữ liệu 8 bits

3. RS (Register Select) - chọn thanh ghi:

Có hai thanh ghi rất quan trọ ng bên trong LCD. Chân RS được dùng để ch ọn các thanh ghi này. Nếu RS = 0 thì thanh ghi mã lệnh được chọn, cho phép ngườ i dùng gửi mộ t l ệnh chẳng hạn như xoá màn hình, đưa con trỏ về đầu dòng v.v…Nếu RS = 1 thì thanh ghi dữ liệu được chọn và cho phép người dùng gửi dữ liệu cần hiển thị lên LCD.

4. R/W (Read/Write) – chân đọc/ghi:

Chân vào đọc/ghi cho phép người dùng đọc/ghi thông tin từ/lên LCD. R/W = 0 thì đọc, R/W = 1 thì ghi.

5. E (Enable) – chân cho phép:

Chân E được LCD sử dụng để chốt thông tin hiện có trên chân dữ liệu. Khi dữ liệu đượ c cấp đến chân d ữ liệu thì một xung mức Cao -xuống-thấp được áp đến chân E để LCD chốt dữ liệu trên chân dữ liệu. Xung này phải rộng tối thiểu 450ns.

6.D0 – D7:

Đây la 8 chân dữ liệu 8 bits, được dùng để gửi thông tin lên LCD hoặc đọc nội dung của các thanh ghi trong LCD.

Để hiển thị chữ cái và con số, mã ASCII của các chữ cái từ A đến Z, a đến z, và các con số tự 0 – 9 được gửi đến các chân này khi RS = 1.

Cũng có các mã lệnh được gửi đến LCD để xoá màn hình hoặc đưa con trỏ về đầu dòng hoặc nhấp nháy con trỏ.

Cũng có thể sử dụng RS = 0 để kiểm tra bit cờ bận xem LCD đã sẵn sàng nhận thông tin chưa. Khi R/W = 1 và RS = 0 thì cờ bận D7 thực hiện các chức năng như sau: Nếu D7 = 1 (cờ bận bằng 1) có nghĩa LCD đang bận các công việc bên trong và sẽ không nhận bất kỳ thông tin mới nào, còn nếu D7 = 0 thì LCD sẵn sàng nhận thông tin mới. Trong mọi trường hợp cần kiểm tra cờ bận trước khi ghi bất kỳ dữ liệu nào lên LCD.

-LCD ghép nối với các cổng p1.0…..p1.7 cổng p1 là cổng dữ liệu dùng để chuyểndữ liệuvà chênh lệch giá trị , 3chân EN ,RW,RS được nối với các chân của vi xử lý có nhiệm vụ như đã nói ở trên ,chân thứ 15,16 lắp cho đèn LCD (ở đây ta không dùng)

Đo lường và tự động điều khiển

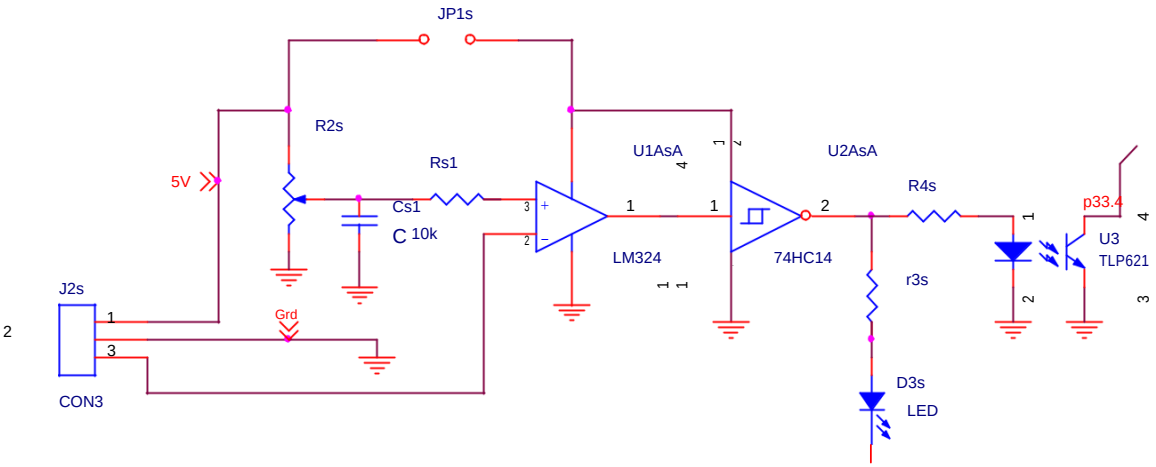
3 .Phần đo động cơ (khối sensor)

Về phần này thì gồm có những bộ phận sau:

+ Bộ thu phát hồng ngoại làm nhiệm vụ thu nhận tốc độ động cơ bằng Diode thu phát quang để đếm số vòng quay của động cơ qua một đĩa chắn quang gắn vào trục của motơ

+ Bộ so sánh ngưỡng LM324 làm nhiệm vụ chuyển tín hiệu từ bộ thu phát hồng ngoại thành tín hiệu TTL tương ứng đưa vào chân P1.0 của vi điều khiển.LM 324 có nhiệm vụ khuyếch đại thuật toán

74HC14 trigosmith hoạt động theo sườn tránh các hoạt động dao động trên đường truyền



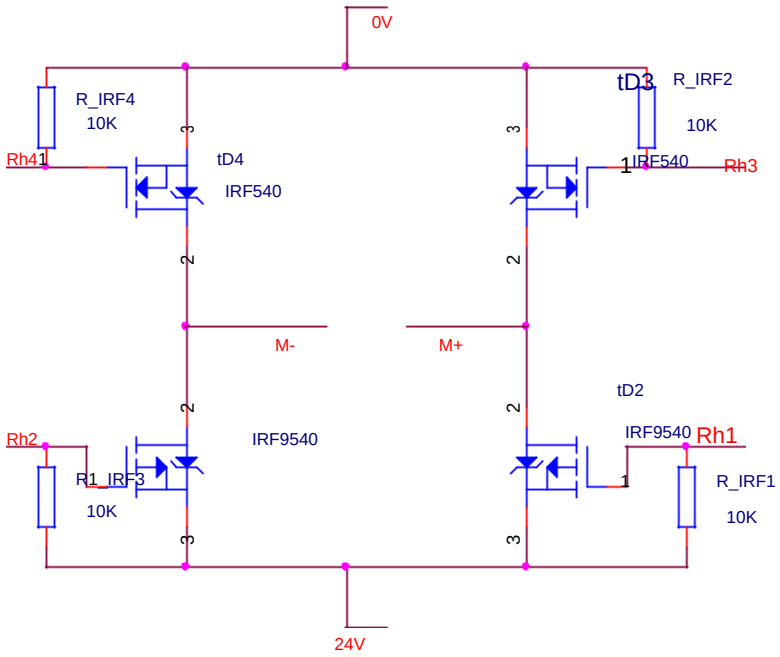
Đo lường và tự động điều khiển

Nguyên lý ho ạt động của mạch: Việc đo tốc độ c ủa động cơ dựa vào quá trình đếm xung của vi xử lí thông qua các xung của bộ sensor thu phát (ứng với mỗ i vòng quay của động cơ sensor sẽ phát 1 xung đưa về bộ đếm của vi xử lí) .
Thông qua chương trình được lập trình sẵn trong vi x ử lí ,mạch đếm sẽ tính toán số li ệu đo được và cho hiển thị kết quả. Với khả năng lập trình được vi xử lí cho phép tự động chọn thang đo (có thể thay đổi khoảng th ời gian trong phép đo t ừ đó thay đổi thang đo theo giây hay phút ...) . Ngoài ra , vi xử lí còn cho phép phát triển thêm các tính năng củ a mạch như điều khiển tố c độ của động cơ khi cần thi ết , lưu giữ các kết quả của các phép đ o trước đó mở rộ ng khả năng ứng dụng trong nhiều lĩnh vực của mạch (đo tần số , đo chu kì...) . Nhờ sử dụng xung động bộ của thạch anh có độ chính xác và ổn định cao nên
vi xử lí cho phép loại bỏ sai số của hệ thống do sự " chậm " của thời điểm "mở" và "đóng" các xung của tín hiệu đi vào mạch tính toán.

4.Phần động lực

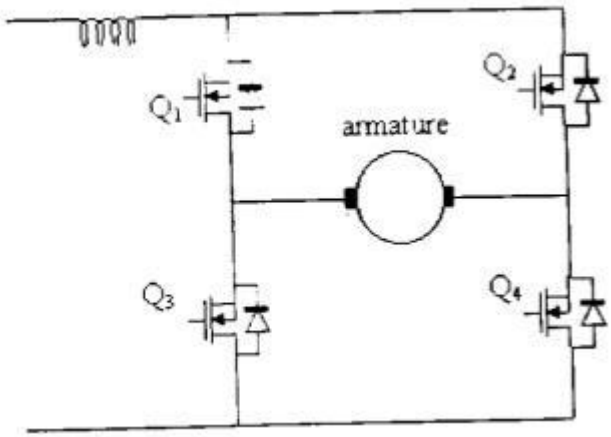
mạ ch cầu H có nhiệ m vụ điều khiển tốc độ động cơ, thay đổi điện áp đầu vào dẫn đến thay đổi thay đổi tốc độ của động cơ

Nguyên lý hoạt động của mạch cầu H:

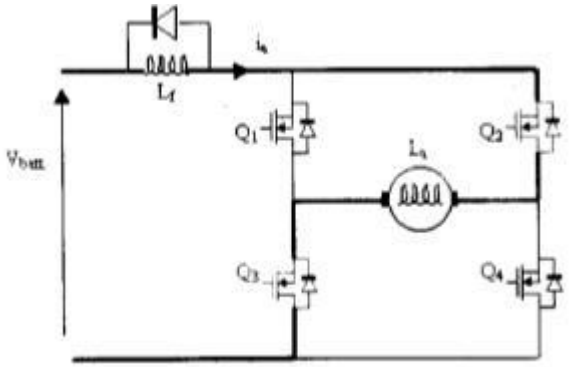


Đo lường và tự động điều khiển

Mạ ch cầu H được mô tả trong hình dưới đ ây. Động cơ đượ c nố i với các cực âm và dương củ a nguồn. Chú ý là chỉ một MOSFET ở mỗi bên của động cơ được bật lên tại một thời điểm nếu không nó sẽ ngắn mạch và cháy.

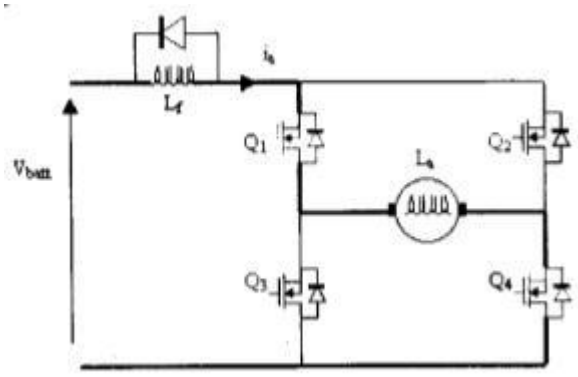


Để động c ơ chạy theo chiều thuận, Q4 được bật, Q1 có tín hiệu điều chế độ rộng xung PWM. Dòng điện được chỉ ra trong hình dưới đây.

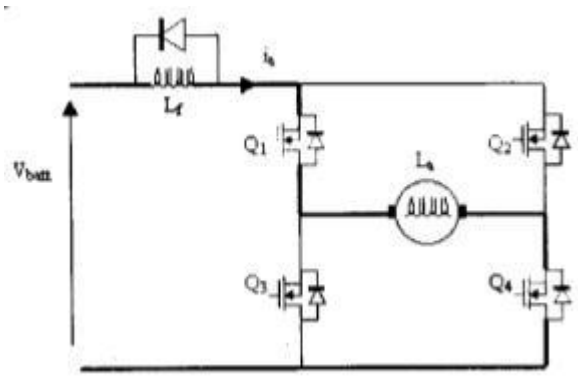


Q4 được giữ sao cho khi tín hiệu PWM mất đi, dòng điện tiếp tục chảy quanh vòng cuối quan đi ốt bên trong của Q3.

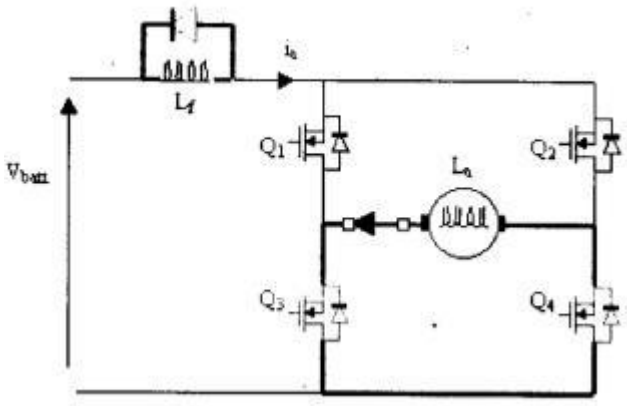
Đo lường và tự động điều khiển



Để động cơ chạy theo chiều ngược, Q3 được bật lên, Q2 có tín hiệu PWM:

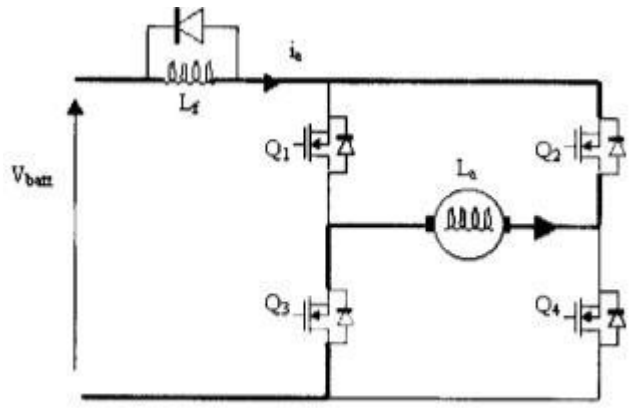


Q3 được giữ sao cho khi tín hiệu PWM mất đi, dòng điện tiếp tục chảy trong vòng cuối của điốt trong Q4:

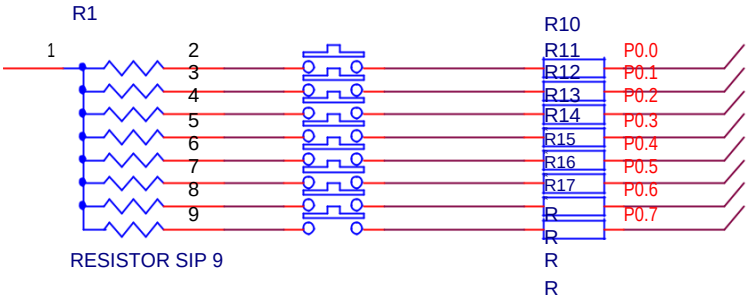


Để phục hồi, ví dụ như khi động cơ đang quay theo chiều nghịch, độ ng cơ (bây giờ đang hoạt độ ng như một máy phát) cho dòng đ iện chảy qua ph ần ứng, qua điốt của Q2, qua nguồn (và do đó nạp cho nó) và quay lại qua điốt của Q3:

Đo lường và tự động điều khiển

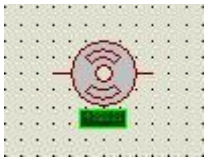


5.Phần nhập giá trị tốc độ ban đầu :



Khối này tương đối đơn giản nó chỉ gồm có các nút bấm hàng đơn vị hàng chục hàng trăm hàng nghìn và hàng vạn ,nút reset ,pause , đảo chi ều và nút start.nó được nối thông qua các điện trở R1 vào cac chân p0.0 đến p0.7 ta nhập các số vào nút bấm dữ liệu.

6.Khối động cơ : H



Hoạt động của độ ng cơ được xác định mỗi khi có sự thay đổI tín hi ệu xung nhận được khi có chùm sáng t ừ cả m biến phát quang chiếu qua khe đạt trên cánh động cơ xuống cả m bi ến thu quang .Tín hiệu thu được từ bộ cảm bi ến hồng ngoạI có tính chất tu ần hoàn do động cơ hoạt động theo chu kì .Chính vì vậy , ta có thể xác định số vòng quay trong 1s.

Đo lường và tự động điều khiển

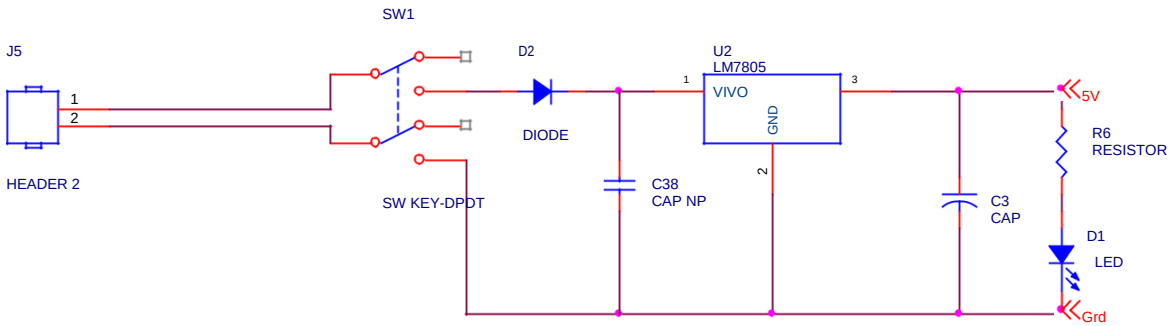
Tín hiệu thu được từ bộ cảm bi ến chưa ổn định do nhiều nguyên nhân khác nhau. Tín hiệu này được đưa vào IC khuếch đạI thuật toán để xử lý .Giá trị điện áp tín hiệu đượ c khuếch đạI lên khoảng 12V tạI đây tín hiệu được đưa vào chân p3.0 chờ xử lý.

Vi xử lý AT89c51 được lập trình vớI đầu vào p3.0 .Port vào ra
+ Port 0 : hàng đơn vị
+ Port 1: hàng chục
+ Port 2: hàng trăm
Đặc điể m n ổi bật của họ vi xử lý 8051 là khả n ăng xử lý dữ liệu theo từng bít .Vì vậy các bít được lập trình sau đó được xuất trực tiếp ra các chân cua LCD mà không cần thông phải qua bộ giải mã 74ls47 .

Chu kỳ lấy mẫu 1 s như vậy tốc độ động cơ được xác định bằng :

Tốc độ động cơ =(tổng số xung /1 s)/6

7.Khối nguồn

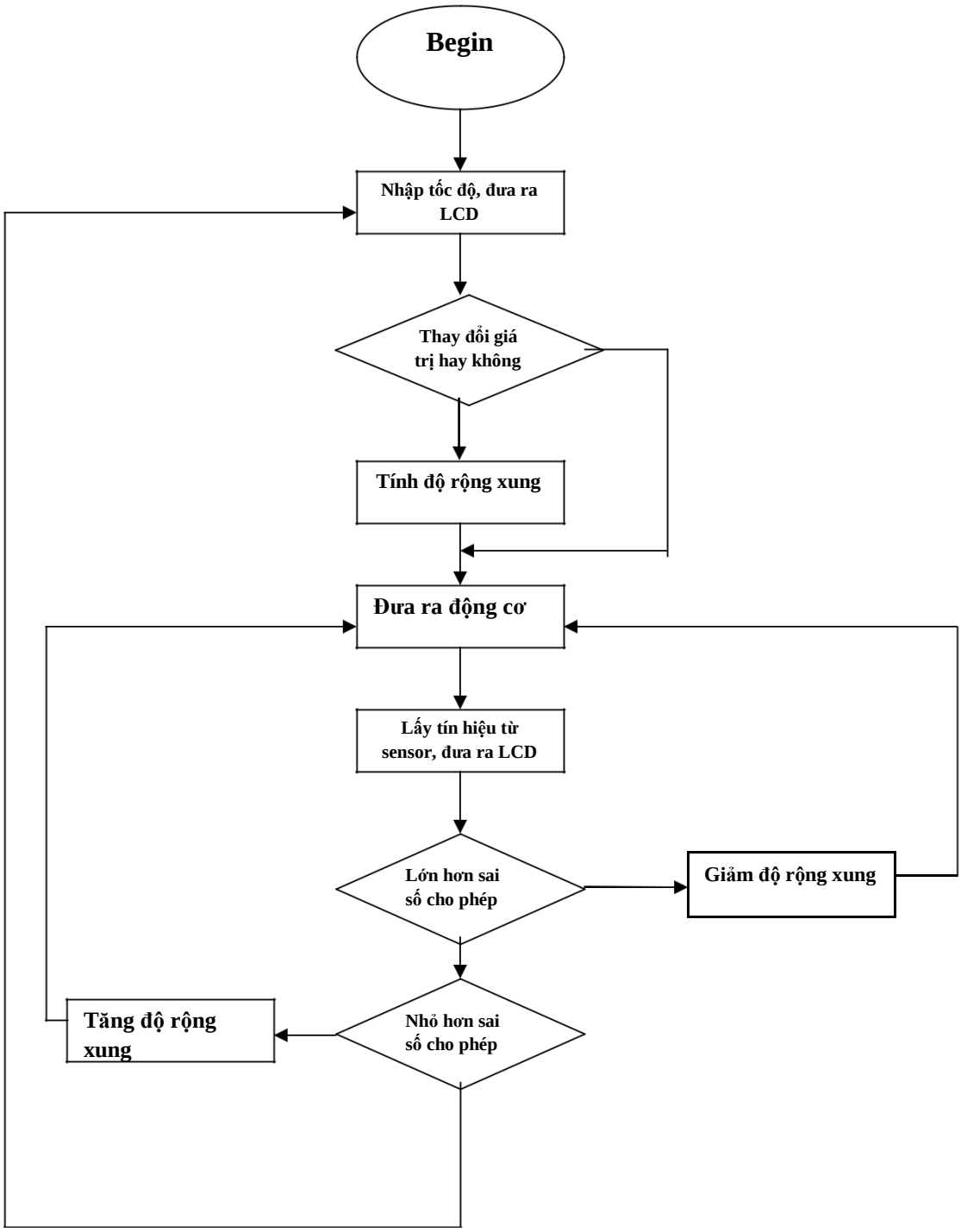


Nguồn điện được thi ết kế riêng s ử dụng IC 7805 cho phép cung cấp điện áp ổn định 5 V với điện áp vào thay đổ i từ 9 –12 V. Mạch nguồn được thiết kế nhằm giảm thiểu sai số do sự không ổ n định củ a điện áp cung cấp cho mạ ch đếm và hạn chế sự ra tăng đi ện áp đột ngộ t gây hỏng linh kiện, cụ thể là 2 tụ C3 và C38 có nhiệm vụ ổn áp tránh dao động do dòng điện, hoặc các hiện tượng sụt áp, ngoài ra con diện trở R6 và con led có nhiệm vụ là bảo vệ nguồn

III, Phần mềm thực hiện thuật toán:

1. Lưu đồ thuật toán.

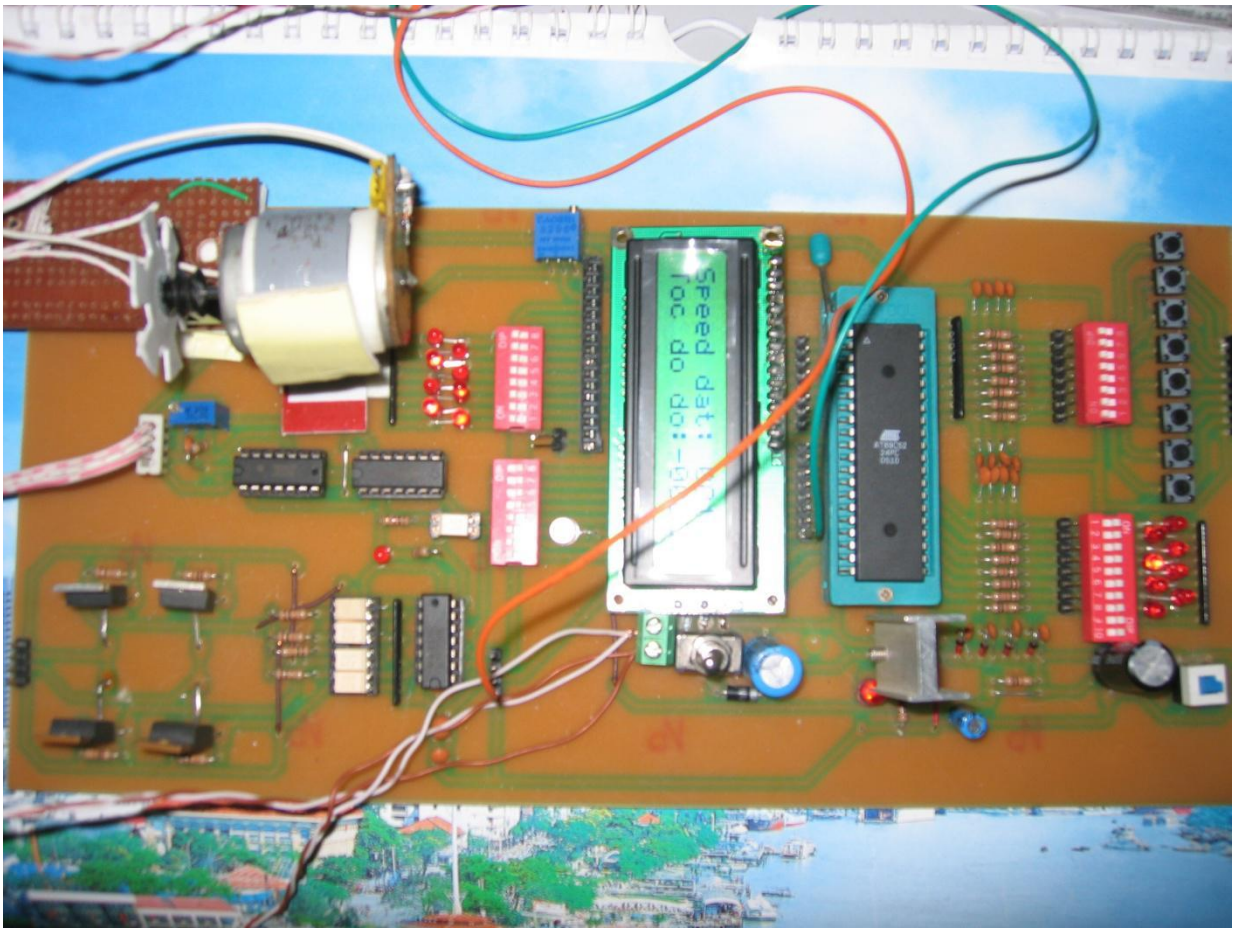
Đo lường và tự động điều khiển



Đo lường và tự động điều khiển

2. Mã nguồn chương trình:

IV .Thiết kế mạch



Sơ đồ mạch in

Đo lường và tự động điều khiển

V. Đánh giá sai số của mạch điện: a)Sai số hệ thống:

a1. Sai số do linh kiện.

Mạch điện có sử dụng một số linh kiện điện tử , các linh kiện này trong điều kiện thường đều có các sai số.

VD : trong vi xử lí 89c51 mỗi câu lệnh đều yêu cầu một thời gian trễ nhất định để thực hiện thời gian này lại phụ thuộc vào xung của bộ dao động thạch anh (có sai số trong quá trình sản xuất) điều này gây ảnh hưởng đến thời gian đếm các xung không còn chính xác như với hàm Delay : bộ định thời làm việc với tần số bằng 1/12 tần số thạch anh tức là bằng

12/12= 0.1 Mhz . Kết quả là mỗi nhịp xung đồng hồ có chu kỳ

T = 1/f = 1 Ms như vậy bộ đếm sẽ tiến hành đếm tăng sau mỗi chu kỳ T với độ trễ được tính = số đếm * 1 Ms.

Ngoài ra khi tín hiệu qua bộ khuếch đại và qua cổng NOT có sự trễ do thời gian đóng mở của linh kiện dẫn đến sự mất đồng bộ về thời gian gây sự chuyển trạng thái của tín hiệu một cách ngẫu nhiên gây sai lệch trong phép đếm xung (còn gọi là sai số +1).

a2. Sai số của nguồn tín hiệu

Nguồn tín hiệu được tạo ra từ mạch sensor thu phát hồng ngoại . Mức điện áp khi có tín hiệu (đã qua khuếch đại) là 3,5 - 4 V còn khi không có tín hiệu là 0- 2,5 V .Tuy nhiên trong quá trình thu phát tín hiệu do động cơ quay với tốc độ khá cao dẫn đến sự chuyển mức không kịp đáp ứng tạo nên một dãy các xung ứng với mức “1” hoặc “0” mặc dù mức điện áp không nằm trong các mức này. Đây chính là nguyên nhân chính gây ra sai số của mạch điện.

a3. Sai số do cách xử lí kết quả đo

Đo lường và tự động điều khiển

Trong quá trình đ o chưa tính đượ c các thông số kí sinh của mạch. Trong chươ ng trình chưa xem xét đến độ trễ của các câu lệnh, chưa tìm được cách xử lí độ nhiễu của tín hiệu.

Mạch tạo nguồn tín hiệu bằng sensor , mức tín hiệu chuẩn trong khoảng cách 5cm ,độ ổn định chưa cao còn chịu ảnh hưởng của nguồn sáng bên ngoài (ánh sáng tự nhiên , ánh sáng đèn...).

Trong bài tập này ánh sáng phòng có ảnh hưởng lớn đến sai số trong phép đo vì nó can nhiễu đến s ự thu nhận của sensor . Tuy nhiên điều này được hạn chế bằng mạch khống chế độ nhạy sensor .

b. Sai số ngẫu nhiên.

Trong qua trình đo động cơ có độ rung nhất định (tuỳ thuộc vào mức đi ện áp cung cấp) dẫn đến sự sai khác trong việc thu và xử lí tín hiệu của sensor. Điều kiện bên ngoài cũng có ảnh hưởng đến mạch : độ sáng , độ ẩm...

Nguồn điện cung cấp cho mạch không ổn định dẫn đến điều kiện làm việc của các thiết bị không còn chính xác.

Bên cạnh đó trong quá trình đo còn có sai số do chủ quan của người đo : không giữ đúng khoảng cách sensor thu phát , không cẩn thận trong qua trình đo .

5. ***Một số kết quả đo được***

Với nguồn cung cấp cho động cơ : 12V

lần 1 : nhập giá trị đầu vào : 5999 vòng /1ph
giá trị đo được : 5975vòng /1 ph
lần 2 : : 3500 vòng / 1ph
3494vòng /1ph
lần 3 : 2000 vòng / 1ph
1980 vòng/1ph

Sai số tương đối : 0.5%

VI. Kết Luận

1. Nhận xét về mạch:

a. Ưu điểm:

Mạch đơn giản, dễ lắp đặt sửa chữa,chi phí thấp.

- Mạch cho phép đo được tốc độ của nhiều loại động cơ,với khoảng giới hạn lớn - Mạch có độ chính xác tương đối, có khả năng phát triển thêm các chức năng (kết nối máy tính,...). Có khả năng ứng dụng trong thực tế: Như trong việc hiển thị tốc độ của một số loại máy ,điều khiển được tốc độ của máy móc theo ý muốn của người sử dụng.

Mạch chạy tương đối ổn định ,dùngphối hợp trở kháng và mạch cầu H

Đo lường và tự động điều khiển

Mạch sử dụng các đot phát quang dựa trên sự tái hợp của điện tử và lỗ trống của lớp tiếp xúc PN.

b. Khuyết điểm:

- Mạch còn sai số do linh kiện:
- Chịu ảnh hưởng của nhiễu do ánh sáng đến khối thu phát của sensor.
- Sensor chỉ cho phép đo trong khoảng cách tương đối ngắn.

2. Phương án cải tiến.

- Cải thiện khả năng của khối thu phát: nâng cao khả năng chống nhiễu cũng như khả năng thu phát của sensor (tăng khuyếch đại, sử dụng sensor thu phát trực tiếp).
- Thêm một số tính năng :

+ Sử dụng phương pháp điều khiển tốc độ bằng phương pháp thay đổi độ rộng xung PWM.
+ Thêm khả năng lưu giữ các giá trị đa đo khi cần có thể xuất ra hiển thị bằng các nút bấm.

VII.Xu hướng phát triển của đề tài

-Sản phẩm có th ể được phát triển thêm các chức năng đo khác như đo nhiệt độ, độ ẩ m ,áp suất và điều khiển, khống chế được .Hơn nữa các chức năng có thể liên quan đến nhau bổ trợ và kiểm soát lẫn nhau theo yêu cầu của người sử dụng

-Sản phẩm sẽ phát triển thành mộ t máy có th ể đo các loại động cơ lớn hơn ,đồng thời nhưng có hướng xử lý khác nhau cho t ừng loại tốc độ ,từng thời gian khác nhau ta có thể phát triển thành một máy đo được cài và thực hiện mọi chương trình được nhập vào máy, trong một quy trình công nghiệp

LỜI KẾT

Qua thời gian 10 tuần làm bài tập lớn là môt khoảng thoig gian ngắn nhưng cũng đã giúp chúng em hiểu thêm về việc thiết kế một hệ thống trong công nghiệp có sử dụng vi điều khi ển đặc biệt là vi đi ều khiển 8051. Trong công nghiệp hiện nay việc đưa các vi điều khiển vào để tạo ra các thiết bị thông minh là cần thiết. Từ đó các hệ thống được tao ra sẽ gon nhẹ hơn và giải quyết nhiệm v ụ nhanh hơn, dễ dàng hơn. Trong thiết kế hệ thống này đã đáp ứng được yêu cầu của bài toán đặt ra, vận dụng tốt những tính năng ưu việt của vi điều khiển 8051 cũng như kết nối các thiết bị được sử dụng một cách hợp lý.

Đo lường và tự động điều khiển

Nh ờ việc làm bài tập lớn nay ma chung em có hiểu được nhưng phần lý thuyêt đ ã được học. Chúng em đã hoàn thành việc thiết kế đo và điều khi ển động cơ một chiều loại nh ỏ. Nh ưng do thời gian có hạn và kiến th ức thực tế chưa có nên vi ệc tìm hiểu cũng như vận dụng còn nhiều hạn chế, chúng em rất mong được sự góp ý của các thầy để mạ ch thiết kế được hoàn thiện hơn. Chung em rất cám ơn sự giúp đỡ nhiệt tình của các bạn cùng lớp dặc biệt là thầy Tuân –thầy giáo dạy môn Đo lường điều khiển t ự động -đã truyền đạt những ki ến thức gắn gũi và tạo điều kiện hết sức cho chung em hoàn thành đ ược bài tập này.

VIII. **Tài liệu tham khảo**

1 - Kỹ thuật mạch điện tử ..... Phạm Minh Hà.

2 - Kỹ thuật số ..... Nguyễn Thúy Vân.

3 - Kỹ thuật đo lường điện tử ..... Vũ Quý Điềm.

4- Cấu trúc và lập trình vi điều khiển 8051……Nguyễn Tăng Cường
Phan Quốc Thắng.
5- Lập trình họ vi điều khiển 8051……..Tống Văn On.