EMG Controller

Dec.2nd, 2005
Kojiro Matsushita

生体信号である筋電位(EMG)を用いたＲＣサーボモータ制御電子回路を紹介する．この電子回路は，全波整流部，RC ローパスフィルタ部，そして AKI-H8/3664 によるＲＣサーボ駆動部の三構成となっている．具体的には，全波整流部＆ＲＣローパスフィルタ部で EMG から強弱信号を抽出し，AKI-H8 においてその強弱信号をＲＣサーボモータの角度に対応させている．

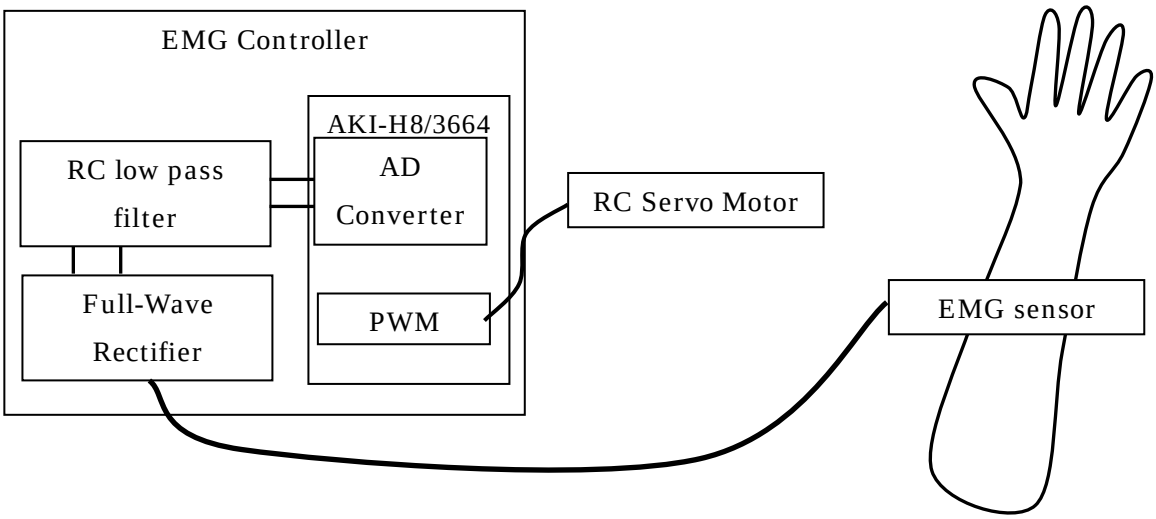

Fig. Basic Concept

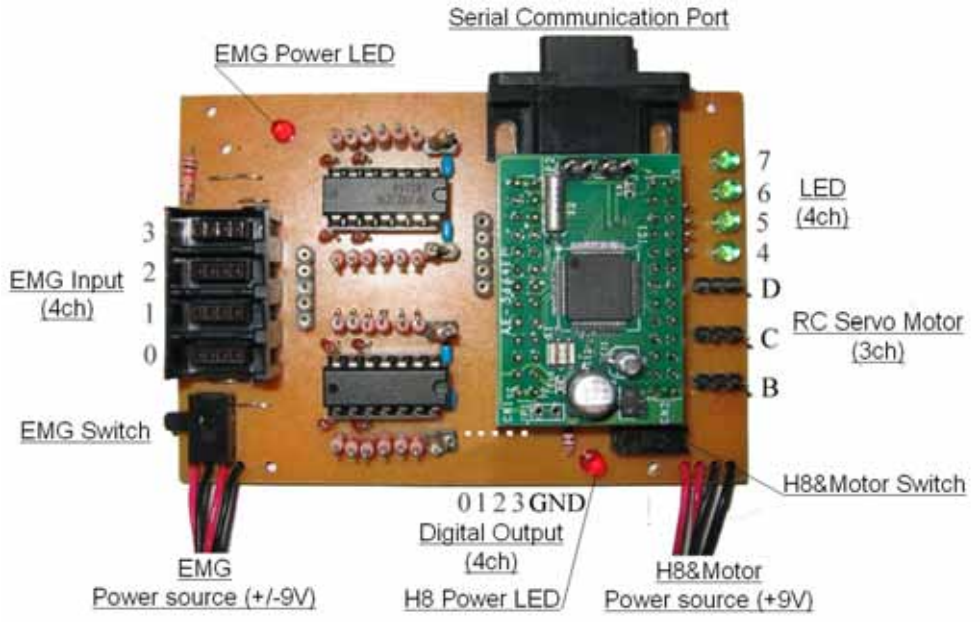

Fig Electric Circuit

Table Correspondence between electric circuit and H8/3664 function

Controller	H8/3664	H8 programming
EMG sensor 0ch	AD0ch	x0 (AD.DRA)
EMG sensor 1ch	AD1ch	x1 (AD.DRB)
EMG sensor 2ch	AD2ch	x2 (AD.DRC)
EMG sensor 3ch	AD3ch	x3 (AD.DRD)
LED7	Port1: 7pin	IO.PDR1.BIT.B7
LED6	Port1: 6pin	IO.PDR1.BIT.B6
LED5	Port1: 5pin	IO.PDR1.BIT.B5
LED4	Port1: 4pin	IO.PDR1.BIT.B4
Digital Output3	Port5: 3pin	IO.PDR5.BIT.B3
Digital Output2	Port5: 2pin	IO.PDR5.BIT.B2
Digital Output1	Port5: 1pin	IO.PDR5.BIT.B1
Digital Output0	Port5: 0pin	IO.PDR5.BIT.B0
RC Servo Bch	Port8: 2pin	TW.GRB
RC Servo Cch	Port8: 3pin	TW.GRC
RC Servo Dch	Port8: 4pin	TW.GRD

EMG Sensor

Amplification:1000*80=80000

Voltage output: 0.2V (approx. min.) to 2.4V (approx. max.)

RC Low pass filter:

R=180[kohm], C=0.1[uF], Cut-off Frequency=8.8[Hz]

RC servo:

TW.GRB=1400(Approx. 0 deg.) to TW.GRB=4600(Approx. 180 deg.)

EMG signal to RC servo

TW.GRB=1400+(x0-10)*GAIN

10 is initial voltage, GAIN(=30) is a coefficient of amplification of x0 (from 0 to 255).

1400 is initial position of RC servo.

LED(4ch) & Digital Output(4ch)

EMG Voltage (0-5V)	AD0ch (0-255)	LED(4ch) & Digital Output(4ch)							
		Port1				Port5			
		7	6	5	4	3	2	1	0
0V-0.6V	0<=x0<30	0	0	0	0	0	0	0	0
0.6V-1.2V	30<=x0<60	1	0	0	0	0	0	0	1
1.2V-1.8V	60<=x0<90	1	1	0	0	0	0	1	1
1.8V-2.4V	90<=x0<120	1	1	1	0		1	1	1
2.4V-4.0V	120<=x0<=200	1	1	1	1	1	1	1	1
4.0V- (error)	x0>200	0	0	0	0	0	0	0	0

Appendix: emgcntrl.c

```
#include <3664f.h>
void main(void){
        unsigned char buf;
        unsigned char x0,x1,x2,x3;      /* AD 値 0ch-3ch */
        int cnt=0, t=0;
        int bpos, cpos, dpos, smb[10], smc[10], smd[10];
        int i, initb, initc, initd;     /* RC サーボの最低角度 */
        int gain=30;                    /* AD 値から RC サーボへのゲイン */

        initb=1400;
        initc=1400;
        initd=1400;

        bpos=initb;
        cpos=initc;
        cpos=initd;
        for(i=0;i<10;i++){
                smb[i]=initb;
                smc[i]=initc;
                smd[i]=initd;
        }

        IO.PMR1.BYTE = 0x00;        /* ポート１を汎用ＩＯに設定 */
        IO.PMR5.BYTE = 0x00;        /* ポート 5 を汎用ＩＯに設定 */
        IO.PCR1 = 0xff;             /* ポート１を出力に設定 */
        IO.PCR5 = 0xff;             /* ポート 5 を出力に設定 */

        TW.TMRW.BIT.PWMB=1;         /* PWMB を設定                */
        TW.TMRW.BIT.PWMC=1;         /* PWMC を設定                */
        TW.TMRW.BIT.PWMD=1;         /* PWMD を設定                */
        TW.TCRW.BIT.CCLR=1;         /* TCNT が GRA でクリア       */
        TW.TCRW.BIT.CKS=3;          /* クロックφ/8                */
        TW.TCRW.BIT.TOB=1;          /* PWMB:最初の出力値          */
        TW.TCRW.BIT.TOC=1;          /* PWMC:最初の出力値          */
        TW.TCRW.BIT.TOD=1;          /* PWMD:最初の出力値          */

        AD.CSR.BIT.ADST=0;          /* AD 開始            */
        AD.CSR.BIT.SCAN=1;          /* スキャンモード     */
        AD.CSR.BIT.CKS=1;           /* 高速モード         */
        AD.CSR.BIT.CH=3;            /* AD0-3ch                    */

        TW.GRA=40000;               /* パルスの周期 20msec        */
        TW.GRB=bpos;                /* PWMB:パルス幅              */
        TW.GRC=cpos;                /* PWMC:パルス幅              */
        TW.GRD=dpos;                /* PWMD:パルス幅              */
        TW.TMRW.BIT.CTS=1;

        while(1){
                if(TW.TSRW.BIT.IMFA==1){
                        TW.TSRW.BIT.IMFA=0;
                                AD.CSR.BIT.ADST=1;
                                while(AD.CSR.BIT.ADF==0){};
                                AD.CSR.BIT.ADF=0;
                                AD.CSR.BIT.ADST=0;
                                x0=(AD.DRA>>8);
                                x1=(AD.DRB>>8);
```

```
x2=(AD.DRC>>8);
x3=(AD.DRD>>8);

/* LED(PORT1) & Digital Output(PORT5) */
if((x0>=0)&&(x0<30)){
		IO.PDR1.BIT.B7=0;
		IO.PDR1.BIT.B6=0;
		IO.PDR1.BIT.B5=0;
		IO.PDR1.BIT.B4=0;

		IO.PDR5.BIT.B0=0;
		IO.PDR5.BIT.B1=0;
		IO.PDR5.BIT.B2=0;
		IO.PDR5.BIT.B3=0;
}
else if((x0>=30)&&(x0<60)){
		IO.PDR1.BIT.B7=1;
		IO.PDR1.BIT.B6=0;
		IO.PDR1.BIT.B5=0;
		IO.PDR1.BIT.B4=0;

		IO.PDR5.BIT.B0=1;
		IO.PDR5.BIT.B1=0;
		IO.PDR5.BIT.B2=0;
		IO.PDR5.BIT.B3=0;
}
else if((x0>=60)&&(x0<90)){
		IO.PDR1.BIT.B7=1;
		IO.PDR1.BIT.B6=1;
		IO.PDR1.BIT.B5=0;
		IO.PDR1.BIT.B4=0;

		IO.PDR5.BIT.B0=1;
		IO.PDR5.BIT.B1=1;
		IO.PDR5.BIT.B2=0;
		IO.PDR5.BIT.B3=0;
}
else if((x0>=90)&&(x0<120)){
		IO.PDR1.BIT.B7=1;
		IO.PDR1.BIT.B6=1;
		IO.PDR1.BIT.B5=1;
		IO.PDR1.BIT.B4=0;

		IO.PDR5.BIT.B0=1;
		IO.PDR5.BIT.B1=1;
		IO.PDR5.BIT.B2=1;
		IO.PDR5.BIT.B3=0;
}
else if(x0>=200){
		IO.PDR1.BIT.B7=0;
		IO.PDR1.BIT.B6=0;
		IO.PDR1.BIT.B5=0;
		IO.PDR1.BIT.B4=0;

		IO.PDR5.BIT.B0=0;
		IO.PDR5.BIT.B1=0;
		IO.PDR5.BIT.B2=0;
		IO.PDR5.BIT.B3=0;
}
else {
```

```
                IO.PDR1.BIT.B7=1;
                IO.PDR1.BIT.B6=1;
                IO.PDR1.BIT.B5=1;
                IO.PDR1.BIT.B4=1;

                IO.PDR5.BIT.B0=1;
                IO.PDR5.BIT.B1=1;
                IO.PDR5.BIT.B2=1;
                IO.PDR5.BIT.B3=1;
            }

            /* AD スレッショルド  */
            if(x0<10){x0=10;}
            if(x1<10){x1=10;}
            if(x2<10){x2=10;}
            if(x3<10){x3=10;}

            /*  スムージング&RC サーボへのマッピング  */
            for(i=0;i<9;i++){smb[i]=smb[i+1];}
            smb[9]=initb+((int)x0-10)*gain;
            bpos=0;
            for(i=0;i<10;i++){bpos=bpos+smb[i];}
            bpos=bpos/10;

            for(i=0;i<9;i++){smc[i]=smc[i+1];}
            smc[9]=initc+((int)x1-10)*gain;
            cpos=0;
            for(i=0;i<10;i++){cpos=cpos+smc[i];}
            cpos=cpos/10;

            for(i=0;i<9;i++){smd[i]=smd[i+1];}
            smd[9]=initd+((int)x2-10)*gain;
            dpos=0;
            for(i=0;i<10;i++){dpos=dpos+smd[i];}
            dpos=dpos/10;

            /* RC サーボの角度制限  */
            if(bpos>=4600){bpos=4600;}
            if(cpos>=4600){cpos=4600;}
            if(dpos>=4600){dpos=4600;}

            TW.GRB=bpos;
            TW.GRC=cpos;
            TW.GRD=dpos;
        }
    }
}
```