Handleiding Mega IvI-bot

Rais Mense
Omar Wit

24 Juni 2011

Omschrijving

De aansturing van de Mega IvI bot werkt over twee IvI-bot prints, deze hebben ieder een eigen taak, en communiceren op eenvoudige wijze over een vijftal poorten.
Er is voor gekozen om de aansturing over twee bordjes te laten verdelen om ervoor te zorgen dat er op beide microcontrolers voldoende ruimte vrij is voor uitbreidingen in de toekomst.

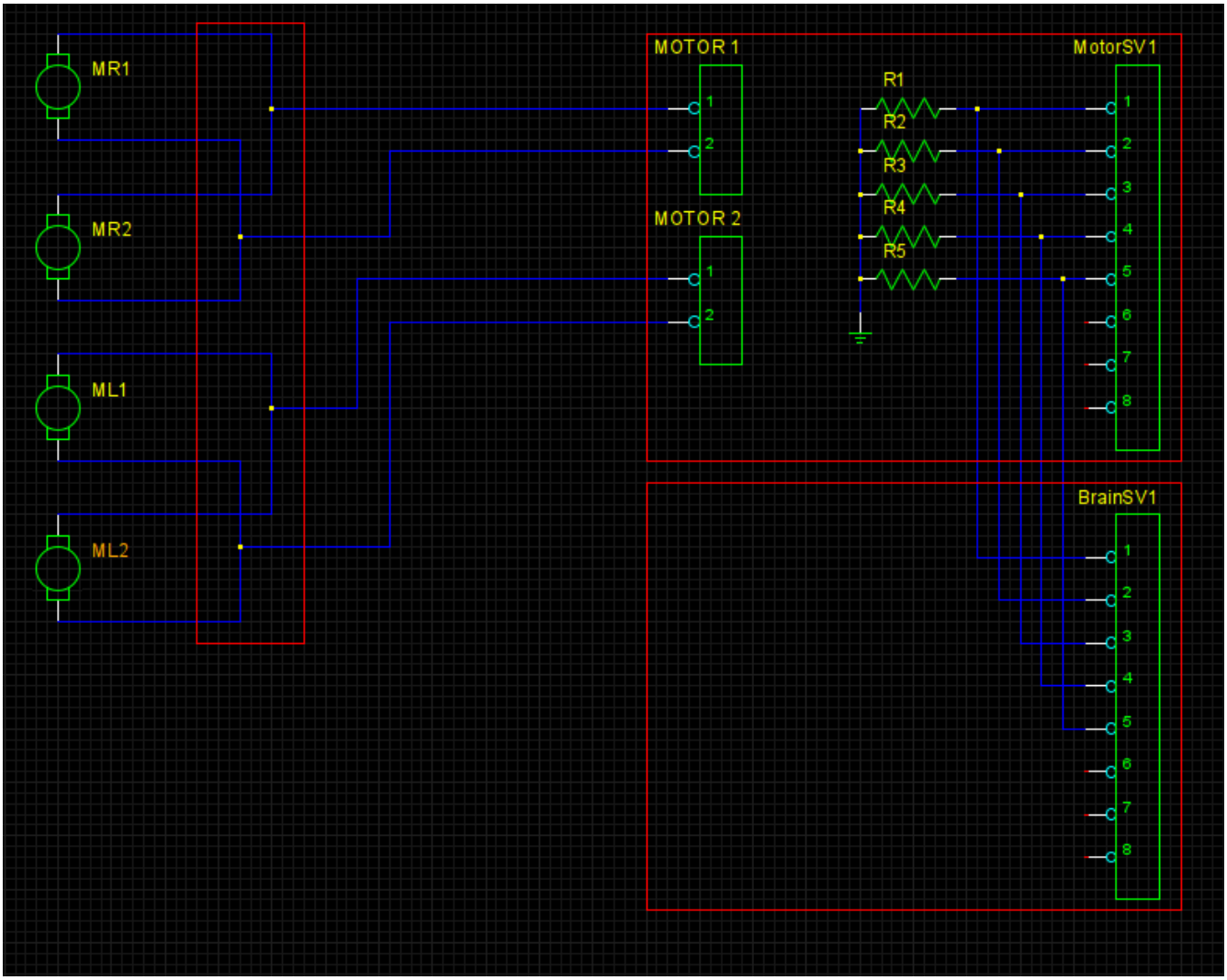
Het 'brains' bordje verzorgd de aansturing en verzend 5 mogelijke commando's over de 5 draden die de bordjes verbinden. Het is mogelijk om in de toekomst minder I/O poorten te gebruiken door een communicatie protocol in te zetten als I^2C, mocht er een tekort aan I/O poorten ontstaan.
Het 'brains' bordje zal uitgebreid kunnen worden voor het verwerken van input.
In de huidige situatie is er ter demonstratie een tweetal arrays in de code opgenomen, de eerste van deze (instructions) bevat integers die verbonden zijn aan een bepaald commando. De lijst van commando's is gedefinieerd in *constants.h*.
De tweede array is ook gevuld met integers, maar in dit geval bepaalt de waarde hoe lang dit commando uitgevoerd moet worden.

Het 'motor' bordje zorgt voor de aandrijving van de wielen door de H-BRUG aan te sturen. Ook zorgt dit bord voor tegen sturing met de tegenovergestelde wielen wanneer er een bocht gemaakt wordt.
Dit bordje kan in de toekomt uitgebreid worden met bijvoorbeeld een ramp-up en ramp-down voor de motor sturing, of wanneer een meer geavanceerd communicatie protocol gebruikt wordt, kan het bordje eventueel ook timing of het nagaan van de gereden afstand op zich nemen.

Schema

Dit schema geeft weer hoe de bordjes en de motoren aangesloten zitten. De twee rode kaders aan de rechter kant geven de IvI-bot prints weer. Het rode kader aan de linker kant is een handgemaakt bordje, dit is enkel om het aansluiten te vergemakkelijken, er kan ook enkel met draad gewerkt worden.

De weerstanden tussen de ground en de verbindingen van voor de aansturings commando's zijn nodig om ruis op de I/O poorten te verhelpen. De waarde van deze weerstanden is niet erg belangrijk en kan tussen de 1K en de 100K Ohm vallen. Hoger en lagere weerstands waarden zijn ook mogelijk, maar niet getest.



Broncode

Beide projecten hebben de benodigede pin definities in een header file staan (*pins.h*). De richting van de I/O wordt in het C bestand gedaan. Per bordje is er een functie aanwezig die dit verzorgt (`void SetMotor( void )` & `void SetBrains( void )`).

Voor het opstarten van de microcontroler zijn bovenaan de C code een aantal regels code aanwezig, deze worden gemarkeerd door `/* Boot code */` en `/* End boot code */`.

In de C code van het Brains bordje wordt gestopt met een `while(1);` Dit is om te zorgen dat het de microcontroler blijft wachten en niet opnieuw begint met het uitvoeren van instructies.

Bootloader

```
/* Boot code */
extern void _startup (void);
#pragma code _RESET_INTERRUPT_VECTOR = 0x001000
void _reset (void) {
    _asm goto _startup _endasm
}
#pragma code
/* End boot code */
```

Deze code is nodig om de microcontroler op te laten starten. Hierna zal de code starten bij de normale `void main(void) { … }`.

Motor aansturing

Voor de aansturing van de motoren zijn er verschillende functies gemaakt. Deze functies zijn te vinden in de motormain.c waar een pin wordt uigelezen en vervolgens de juiste motor wordt aangestuurd.

De aansturing geschied vanuit het brains bordje waar een pin hoog of laag gemaakt wordt, deze pinnen Left, Right, Fwd, Reverse of Stop; worden uitgelezen door het motor-bordje. Bij Left wordt onderstaande code uitgevoerd.

```
if (LEFT) {
    MOTOR_LEFT      = HIGH;
    MOTOR_LEFT_R    = LOW;
    MOTOR_RIGHT     = LOW;
    MOTOR_RIGHT_R   = HIGH;
}
```

Binnen het brains-bordje is onderstaande code voldoende om het wagentje linksom te laten gaan.

```
SetPin(INT_LEFT, HIGH);
```