Wat is firmata?

Firmata is een protocol dat communiceert tussen computers en microcontrollers om gemakkelijk toegang te krijgen tot de Arduino-hardware vanuit software op een hostmachine.

- **Het maakt gebruik van standaard seriële commando's en kan als zodanig op verschillende Arduino-modellen worden gebruikt.**
- **Berichten worden serieel naar en van de hostcomputer verzonden, met vermelding van de pinstatus of met het verzoek om van een pin de status te wijzigen**

Firmata bibliotheek

Het Firmata-protocol heeft verschillende revisies

- **Hoe voorkomen van fouten als twee apparaten verschillende revisies gebruiken?**
  - **specificeren welke protocolrevisie moet worden gebruikt via setFirmwareVersion ():**

setFirmwareVersion (major, minor);

- **De major en minor parameters zijn bytes, die de te gebruiken revisie specificeren.**
  - **Voor de meeste Arduino-toepassingen is dit ingesteld op major versie 0 en minor versie 1.**

Verzenden van berichten

Gebruik maken van de Firmata-bibliotheek kan via oproepen van begin():

Firmata.begin ()

Firmata.begin (snelheid);

Deze functie opent een seriële verbinding.

Standaard is de snelheid ingesteld op 57600 baud,

- **kan worden gewijzigd met de optionele parameter snelheid**

Verzenden van berichten

De status van de pinnen worden verzonden als berichten van- en naar de software op de hostcomputer.

Berichten kunnen worden geadresseerd aan analoge en digitale pinnen

Analoge pin

Vb. Gebruik van sendAnalog(0) om de status van een analoge pin te verzenden:

```
analogValue = analogRead (pin);
Firmata.sendAnalog (pin, analogValue);
```

Verzenden van berichten

Digitale pin (groep pinnen verzenden)

- Zijn in- of uitgeschakeld, om de snelheid langs de seriële verbinding te verhogen wordt de status van een bepaalde pin via de status van digitale pinnen per poort verzonden:

Firmate.sendDigitalPorts(pin, firstPort, secondPort);

- Pinnen moeten in volgorde worden verzonden
  - Bv. Verzenden vanaf pin6 => volgend is pin7, enz …
- firstPort wordt gebruikt om de eerst pin te plaatsen wiens status wordt verzonden
- secondPort wordt gebruikt om het aantal pinnen dat gelijktijdig wordt verzonden in te stellen.

Verzenden van berichten

Digitale pin (status enkele pin of enkele pinnen die niet opeenvolgend zijn)

```
value = digital.Read(pin);
Firmata.sendDigital(pin, value);
```

Verzenden van berichten

Een string verzenden naar de hostcomputer

```
string[ ] = “DSP-cursus”;
Firmata.sendString(string);
```

Dit verzendt de string $string$ naar de hostcomputer

Ontvangen van berichten

Data wordt direct van de seriële poort ontvangen. Om na te gaan of data beschikbaar is kan available() gebruikt worden.

Firmata.available()

Returned true als 1 of meerdere bytes wachten op verdere verwerking/

Ontvangen van berichten

Verwerken van inkomende data:

```
Firmata.processInput();
```

Typisch gebruik:

```
while(Firmata.available( ))
{
Firmdata.processInput( );
}
```

De Firmata-bibliotheek verbergt alle gecompliceerde onderdelen van het ontvangen van gegevens, inclusief de gegevensopslag en -verwerking.

De bibliotheek decodeert automatisch berichten en stelt in staat om acties uit te voeren op de ontvangen gegevens met behulp van een systeem van zogenaamde callbacks.

Callbacks

Callback = routine die aangeroepen wordt wanneer een specifieke actie wordt uitgevoerd (in deze situatie wanneer een bericht wordt ontvangen)

- ✓ ***Zijn zeer aanpasbaar en bruikbaar om bijna elke gewenste actie uit te voeren via het genereren van een functie***
- ✓ ***Worden uitgevoerd via een attach()- functie***

Firmata.attach(messagetype, function);

- ✓ ***Messagetype is een van de gebruikte constanten***
- ✓ ***Function-parameter is de callback-functie die je geschreven hebt.***

Callback-constanten

Constant	Gebruik
ANALOG_MESSAGE	Analoge waarde van een enkele pin
DIGITAL_MESSAGE	Digitale waarde van een digitale poort
REPORT_ANALOG	Schakelt de rapportage van een analoge pin in of uit
REPORT_DIGITAL	Schakelt de rapportage van een digitale poort in of uit
SET_PIN_MODE	Wijzig de modus van de geselecteerde pin (input, output, enzovoort)
FIRMATA_STRING	Gebruikt voor het ontvangen van tekstberichten
SYSEX_START	Gebruikt voor het verzenden van algemene berichten
SYSTEM_RESET	Gebruikt om de firmware terug te zetten naar de standaardstatus

Callbacks (vervolg)

- Callback voor het herstarten van een systeem vereist geen parameters

void.system.ResetCallback(void);

- Callback om strings te ontvangen heeft één parameter nodig

Void stringCallback(char *datastring);

Callbacks (vervolg)

- sysEx-berichten hebben meer info nodig en bevatten 3 parameters
  - System Excusive (oorspronkelijk gebruikt bij synthesizers die het MIDI-protocol gebruiken om gebruikerscommando's te implementeren)
  - Firmata-protocol: sysEx laat gebruikers toe informatie uit te wisselen en instellingen te maken zoals I2C data of een servomotorconfiguratie

void sysex.Callback(byte pin, byte count, byte *array);

- de andere calbacks gebruiken volgend algemeen format:

void generic.Callback(byte pin, int value);

callbacks

- Om een bepaalde pin in te stellen als input of output correspondeert de $mode$ parameter direct met de Arduino pinMode() constanten

- Welke pin komt overeen met wat soort in- of uitvoer?

- ✓Voor elk processorbordje kan men een aantal voorgedefinieerde data gebruiken
- ✓ Het Boards.h bestand definieert hoeveel analoge en digitale pinnen een bepaald controllerbordje bevat
  - ✓Voorbeeld Arduino Uno bordje:

```
#define TOTAL_PINS 20 // 14 digital + 6 analog
```

Gebruik () en IS_PIN_ANALOG 0 om te weten of een pin digitaal is.

Gebruik PIN TO _DIGITAL () en PINTOANALOG () om een pin om te zetten naar een digitaal of analoog equivalent. U kunt de volgende code gebruiken om de status van een digitale pin in te stellen:

void setPinModeCallback (byte pin, int-modus)

callbacks

Gebruik IS_PIN_DIGITAL( ) om te weten te komen of een pin digitaal is.
Gebruik IS_PIN_ANALOG ( ) om te weten te komen of een pin analoog is.

Gebruik PIN_TO _DIGITAL () en PIN_TO_ANALOG () om een pin om te zetten naar een digitaal of analoog equivalent.

Voorbeeld: instellen status van een pin:

```
36 void setPinModeCallback (byte pin, int-modus)
37 {
38   if (IS_PIN_DIGITAL(pin)
39   {
40     pinModea(IS_PIN_DIGITAL(pin)
41     {
42       pinMode(PIN_TO_DIGITAL(pin, mode);
43     }
44   }
45 }
```

Verwijderen van een callback

Gebruik hiervoor detach( );

Firmata.detach(callback);

Callback parameter is één van volgende constanten

- ***ANALOG_MESSAGE***
- ***DIGITAL_MESSAGE***
- ***REPORT_ANALOG***
- ***REPORT_DIGITAL***
- ***SET_PIN_MODE***
- ***FIRMATA_STRING***
- ***SYSEX_START***
- ***SYSTEM_RESET***

System Excusive (SysEx)

- Het idee voor het invoeren van System Excusive berichten was om informatie uit te wisselen en instellingen te wijzigen die niet op een andere manier toegankelijk zijn.
- In het Firmata-protocol kunnen gebruikers informatie uitwisselen, zoals I2C-data en de servomotorconfiguratie.
- Om SysEx-data te ontvangen moet eerst een SysEx Callback gegenereerd worden

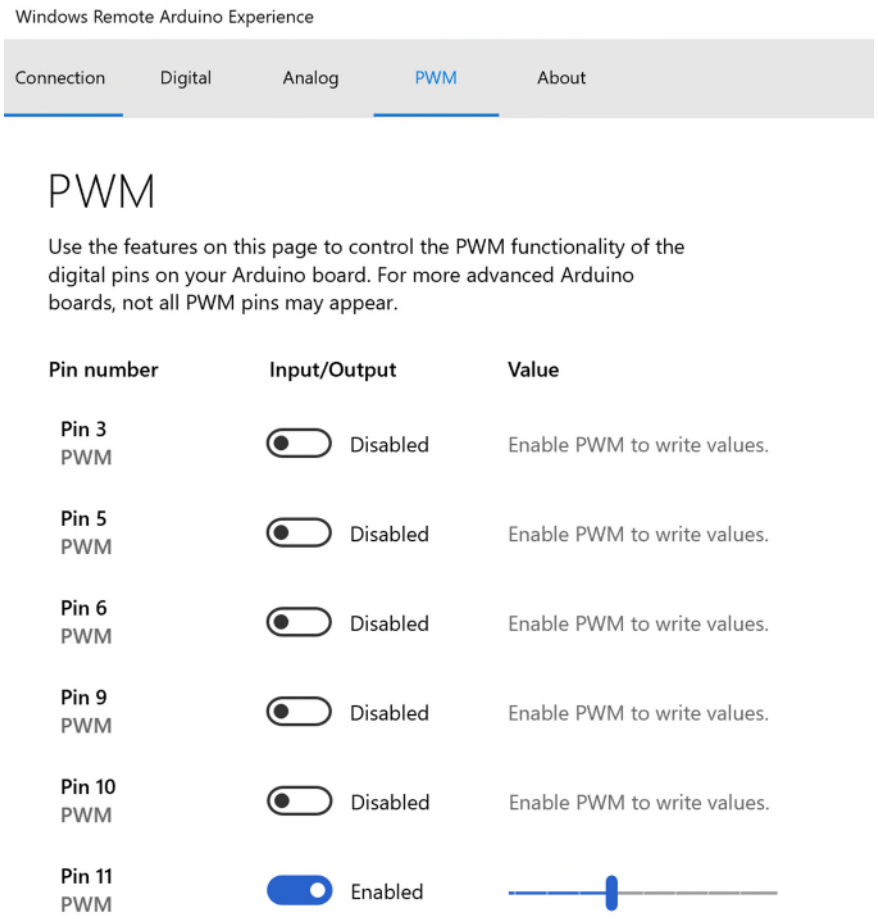
```
void sysexCallback (byte command, byte argc, byte *argv)
{
 // code
}
```

SysEx instructie-ID wordt verzonden als een byte, command genoemd

Constant	Functie	Constant	functie
RESERVED_COMMAND	Gereserveerde chip-specifieke instructies.	STRING_DATA	Zend een stringbericht
ANALOG_MAPPING_QUERY	Vraag voor analoog naar pin-nummer mapping	SHIFT_DATA	34 bits schift-out data
ANALOG_MAPPING_RESPONSE	Antwoord met mappingdata.	I2C_REQUEST	Request PC data
CAPEBILITY_QUERY	Query aangaande ondersteunde modi van alle pinnen.	I2C_REPLY	Antwoord met PC data
CAPEBILITY_RESPONSE	Antwoord met capebility data	I2C_CONFIG	PC parameters
PIN_STATE_QUERY	Vraag naar de huidige mode en waarde van een pin	REPORT_FIRMWARE	Versienummer van Firmata-firmware melden
PIN_STATE_RESPONSE	Atwoord met pinmode en waarde	SAMPLING_INTERVAL	Sample-interval instellen
EXTENDED+ANALOG	Analoog schrijven naar eender welke pin (PWM en servo inbegrepen)	SYSSEX_NON_REALTIME	Voorbehouden voor MIDI
SERVO_CONFIG	Servoparameters instellen (hoek, puls en dergelijke)(	SYSSEX+REALTIME	Voorbehouden voor MIDI

Voorbeeldprogramma

- StandaardFirmata-programma (voor arduino)
  - voorbeeldsketch waarmee je aan de slag kan met Firmata (zit in arduino-IDE onder bestand =>voorbeelden)
- Testprogramma op PC
  - http://www.firmata.org/wiki/Main_Page#Firmata_Test_Prograan (testprogramma voor Windows, Mac OS, and Linux)
- Connectie: USB-poort kennen + baudrate (standard



Voorbeeldprogramma

- Verzenden data van Arduino naar computer via slimme sampletechniek
  - Kan bv LED's in- en uitschakelen zonder een sketch te hoeven schrijven
  - Eenvoudig om je bordje snel uit te testen of bepaalde poorten nog werken of niet.
  - Invoerlijnen lezen vanop afstand

Voorbeeldprogramma

- De standaard Firmata-sketch is een zeer grote sketch (bestaat uit 823 lijnen) en is goed gestructureerd ➔ bruikbaar als basis om een eigen schets te maken

Eigen schets opbouwen:

- In setup () breng volgende regel aan:

```
Firmata.SetFirmwareVersion(Firmata_Major_Version, Firmata_Minor_Version);
```

- Deze lijn stelt de Firmata-versie in, dit is wat Firmata-applicaties controleren
- 2 constanten Firmata_Major_Version en Firmata_Minor_Version worden ingesteld door de Arduino Firmata bibliotheek

Voorbeeldprogramma

- Mogelijke callbacks die je kan instellen in je eigen code binnen setup()

```
Firmata.attach(ANALOG_MESSAGE, analogWriteCallback);
Firmata.attach(DIGITAL_MESSAGE, digitalWriteCallback);
Firmata.attach(REPORT_ANALOG, reportAnalogCallback);
Firmata. attach(REPORT_DIGITAL, reportDigitalCallback);
Firmata.attach(SET_PIN_MODE, setPinModeCallback);
Firmata.attach(START SYSEX, sysexCallback);
Firmata.attach(SySTEM_RESET, systemResetCallback);
```

- Door ze allemal te installeren kan je sketch reageren op elk sort Firmata-bericht (of in ieder geval een specifieke functie aanroepen wanneer het bericht wordt ontvangen)
- Welke je wil installeren hangt van je keuze en code af. Via de voorbeeldsketch StandardFirmata kan je voorbeelden van gebruik bekijken en toepassen in je eigen sketch
- Naast de StandardFirmata sketch vind je onder voorbeelden van de IDE-interface ook eenvoudiger voorbeelden van een specifiek gebruik van het protocol weer

Voorbeeldprogramma

- Het ontvangen en verwerken van de berichten afkomstig van je computer gebeurt in loop()

```
while(Firmata.available())
    Firmata.processInput();
```

Voorbeeldprogramma

- Variabele samplingInterval
  - Bepaalt de snelheid waarmee de Firmata de pinnen peilt
  - Om de gewenste samplefrequentie te behouden wordt volgende code gebruikt:

```
currentMillis = millis ();
if (currentMillis - previousMillis> samplingInterval)
previousMillis + = samplingInterval;
// Code komt hier
```

Voorbeeldprogramma

```
currentMillis = millis ();
if (currentMillis - previousMillis> samplingInterval)
previousMillis + = samplingInterval;
// Code komt hier
```

- *currentMillis en previousMillis zijn unsinged long type variabelenIedere keer Arduino loop() doorloopt ➔ Millis-functie wordt opgeroepen en geeft de waarde in ms weer dat de sketch aan het lopen is*
- *Deze waarde wordt geplaatst in currentMillis*
  - *Als currentMillis – previousMillis > sampleinterval ➔ previousMillis wordt verhoogd met de inhoud van sampleInterval ➔ de sketch is dan vrij om alle pincodegegevens te verzenden*