Variable RF Coupler

Daoud Peter Saadeh

Supervised by:

Nicolas Gandolfo

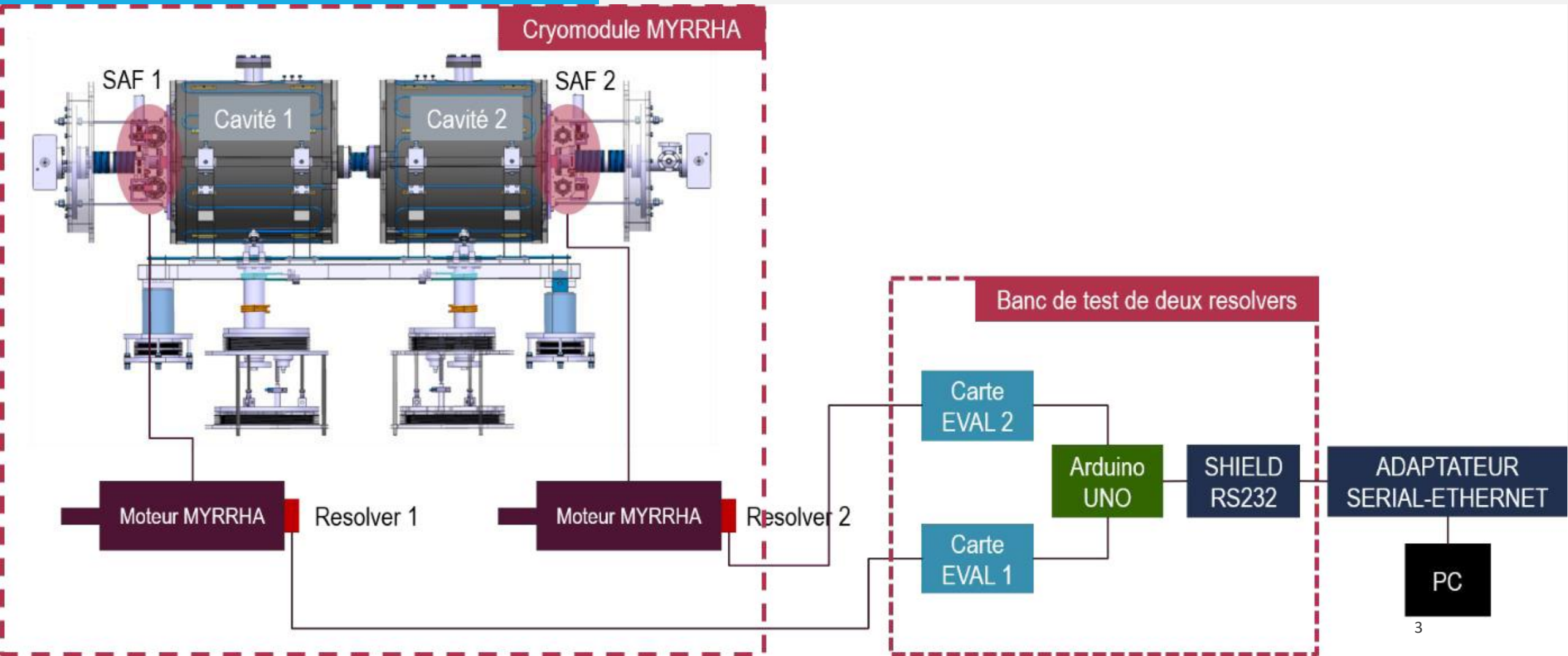
&

David Le Drean

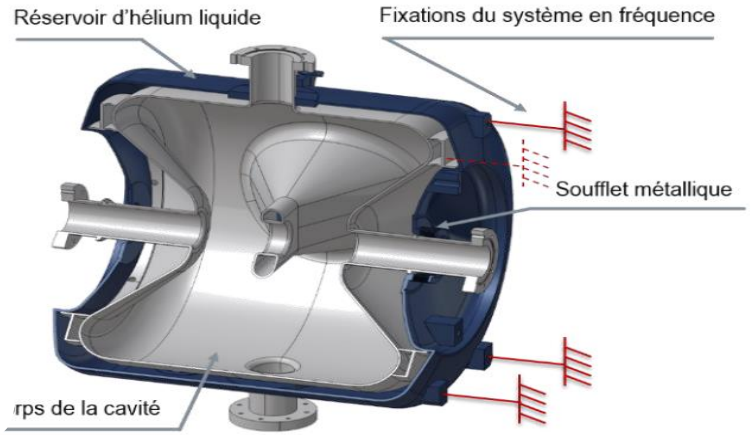
Presentation Outline

- About variable RF coupler
- System Components and Instruments
- Stepper Motor Connection
- LabVIEW Programing
- *Measurement*
- Analysis Outcomes
- Motor Analysis
- Analysis Outcomes
- Current Development Workflow
- System Components and Instruments
- *System integration*
- System Programing Code
- Final system Layout

About variable RF coupler



System Components and Instruments



cavity

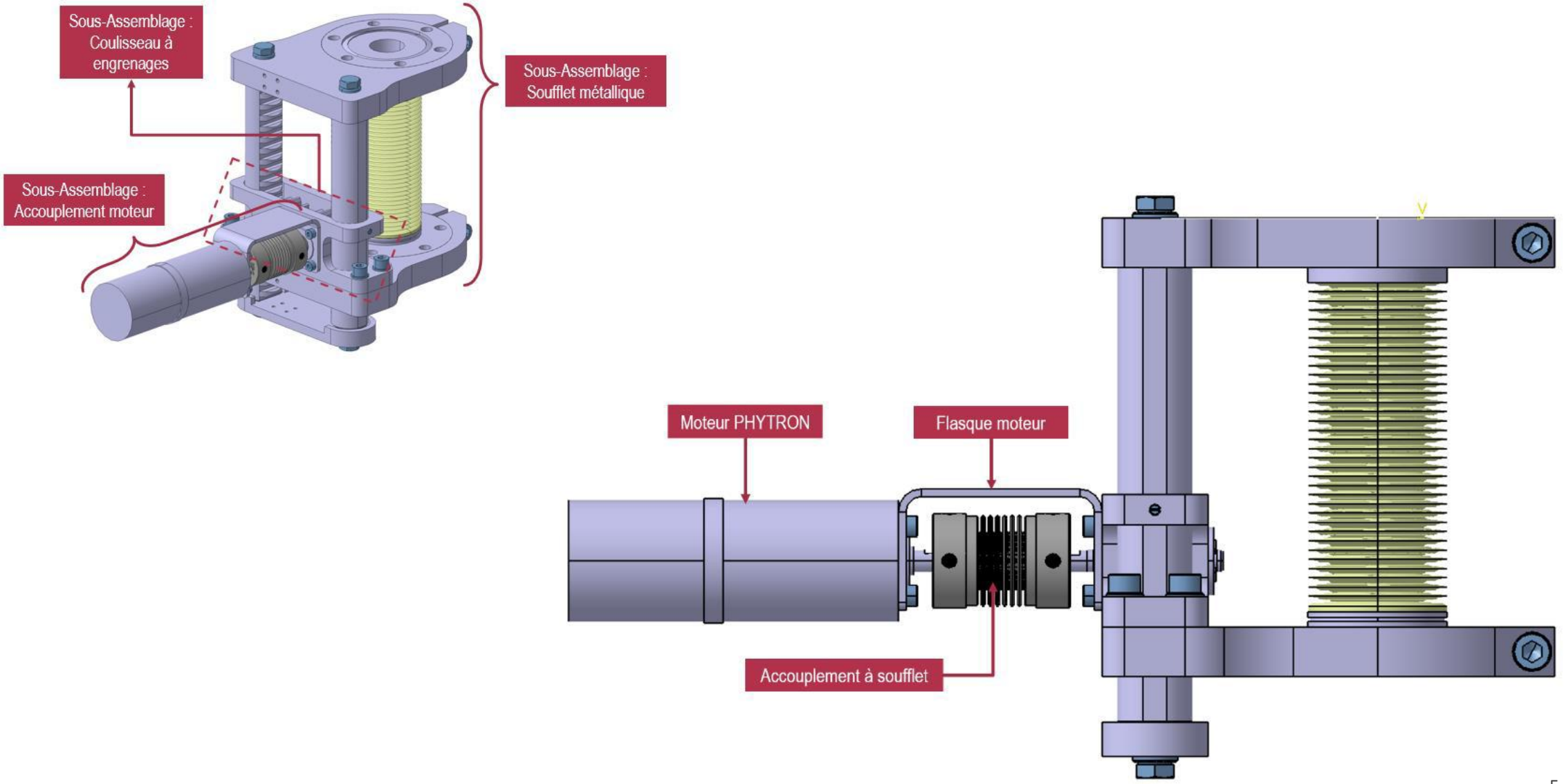
Stepper Motor

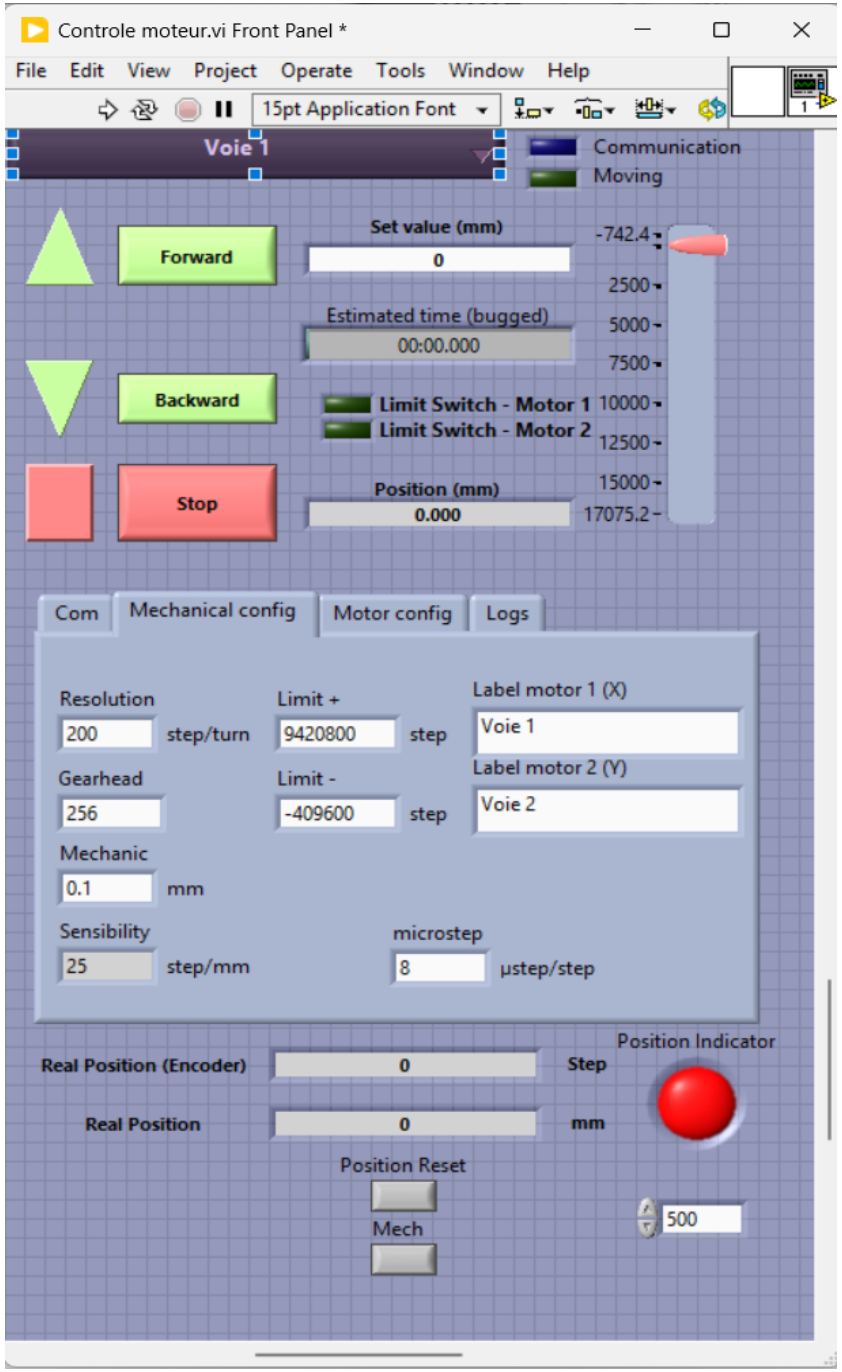
VNA

MCC – 2 Driver

G-REC

Stepper Motor Connection

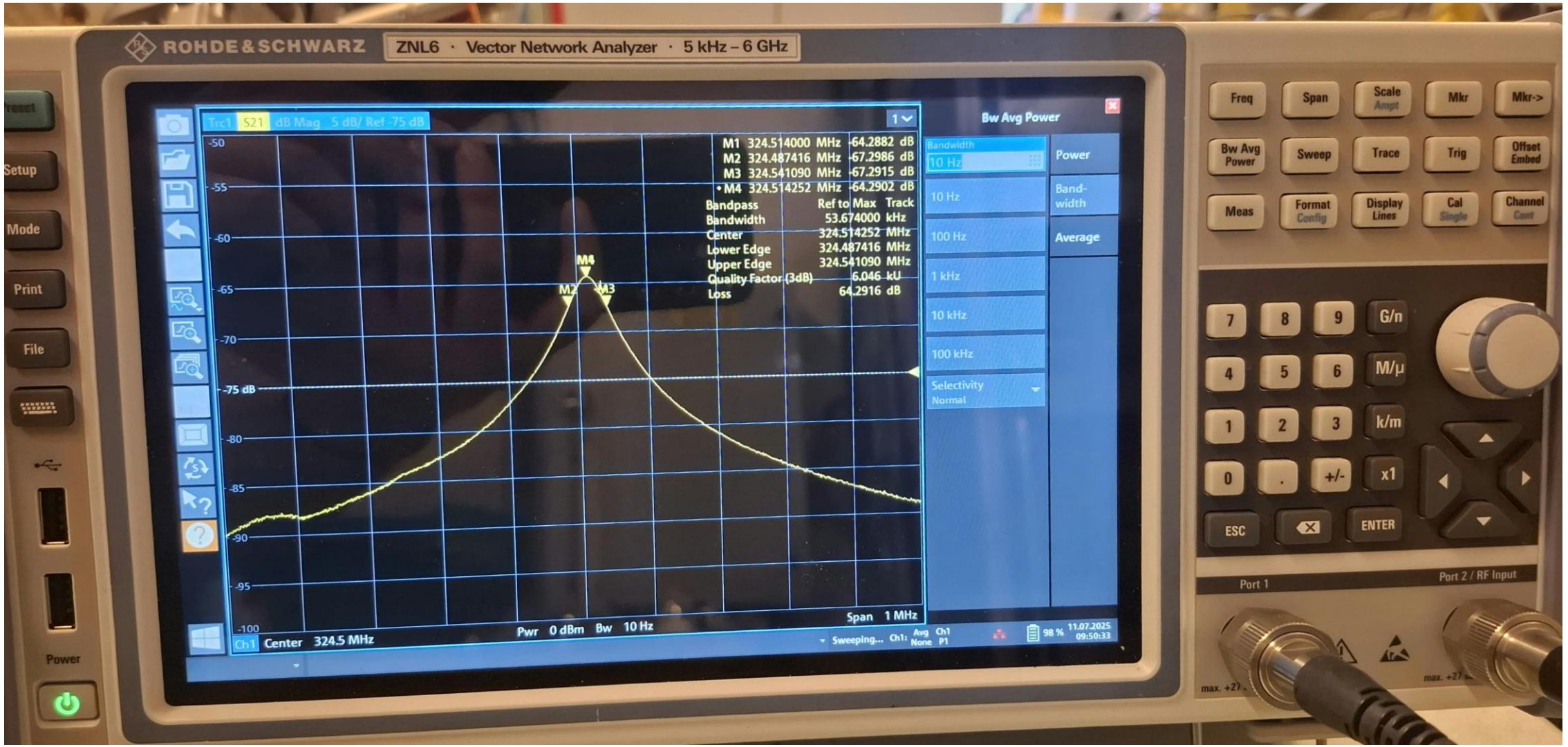




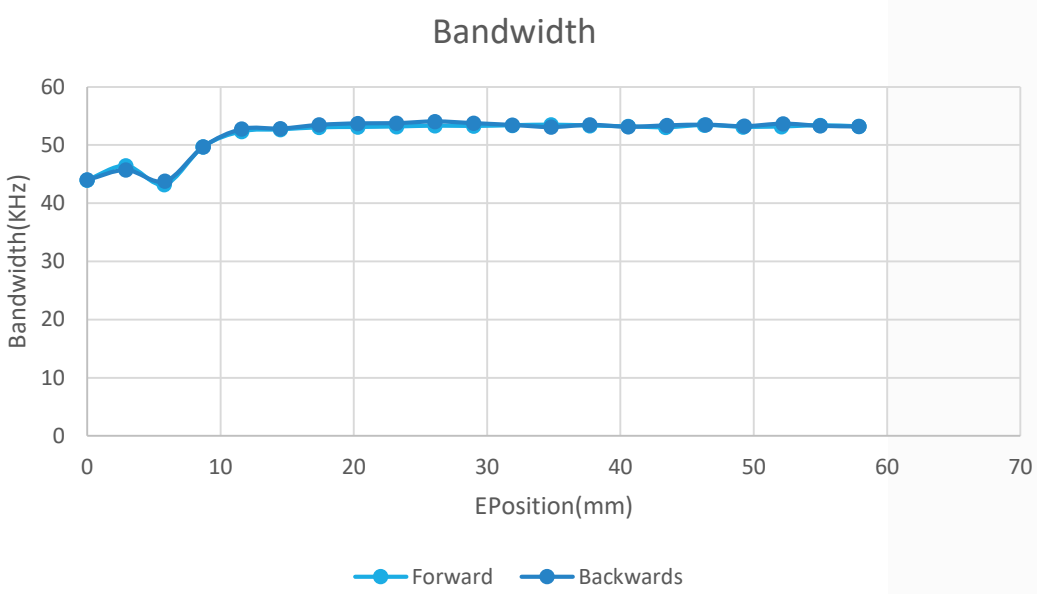
LabVIEW Programing

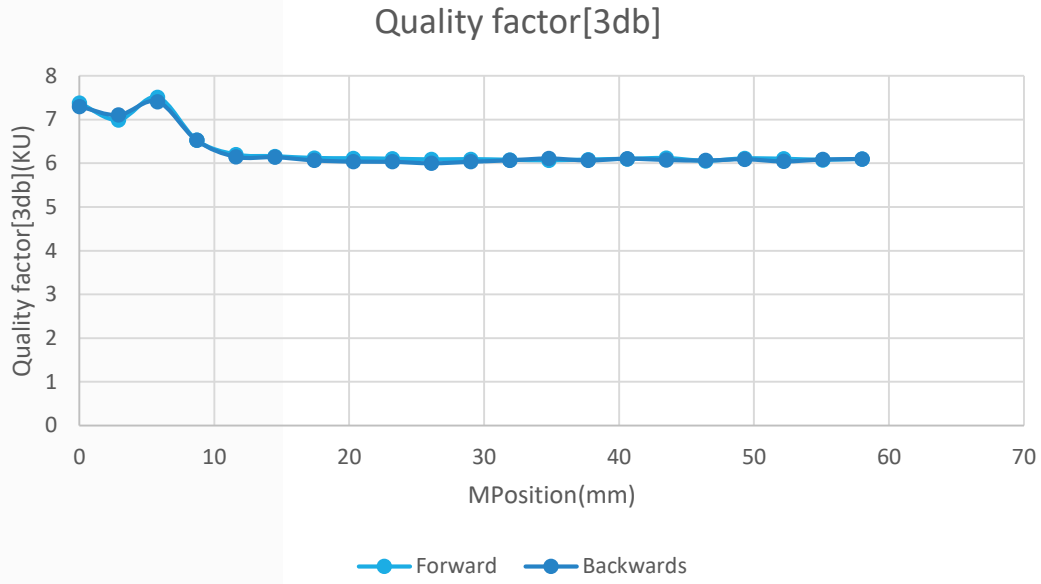
- Customized and developed a pre-existing LabVIEW program.
- Added two indicators to display the motor's actual position using the built-in resolver sensor.
- Implemented a real-time error indicator (LED)
- Added two control buttons

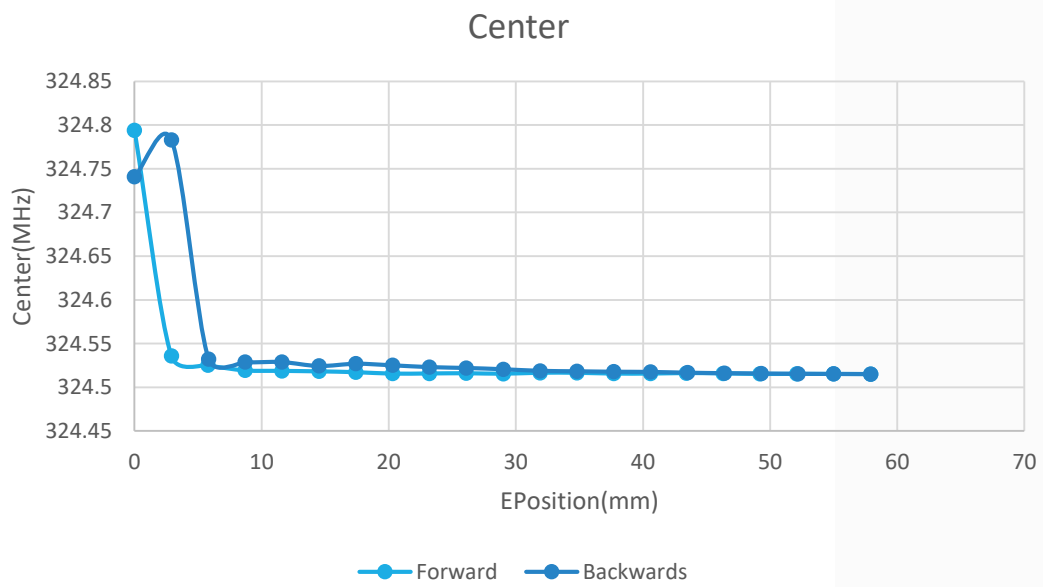
Measurement

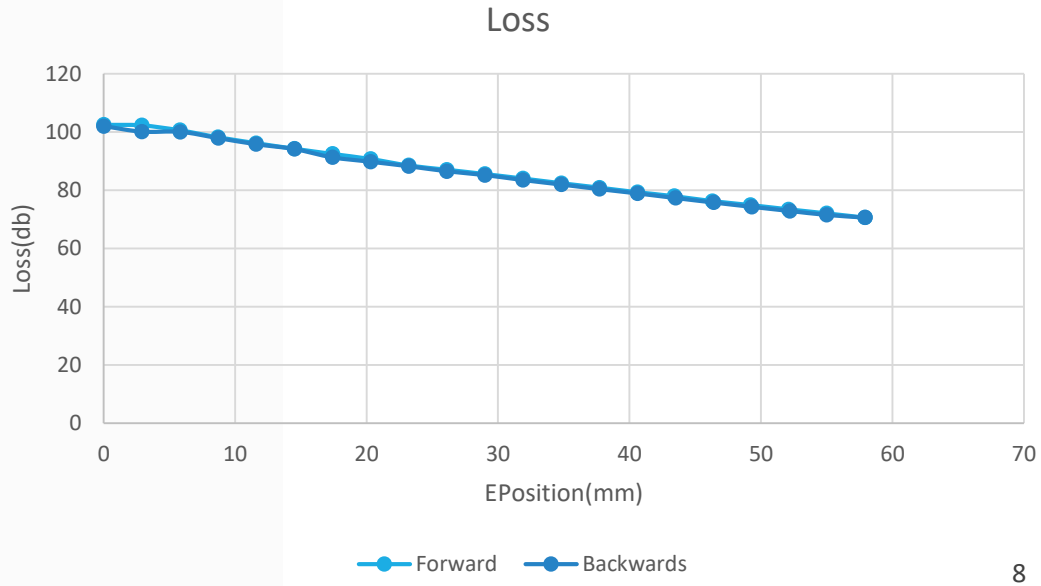


Analysis Outcomes





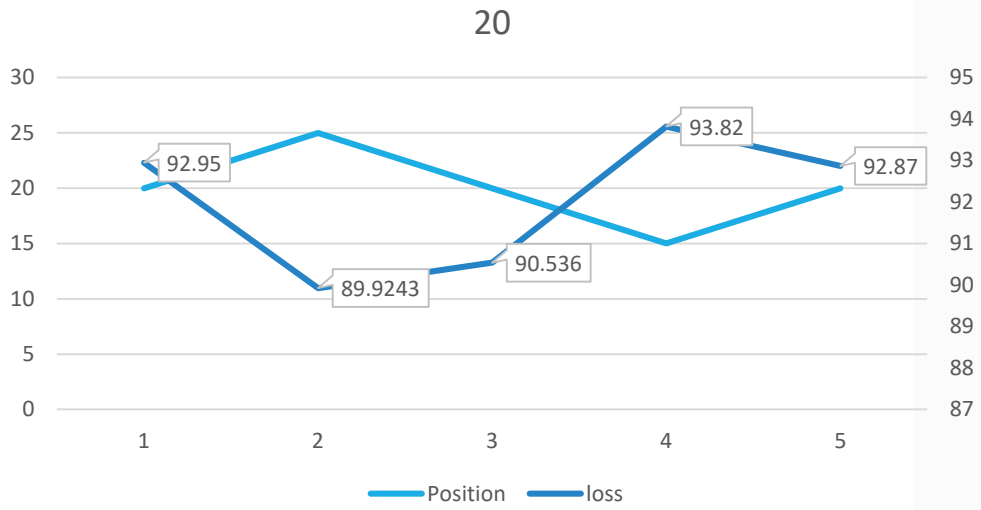


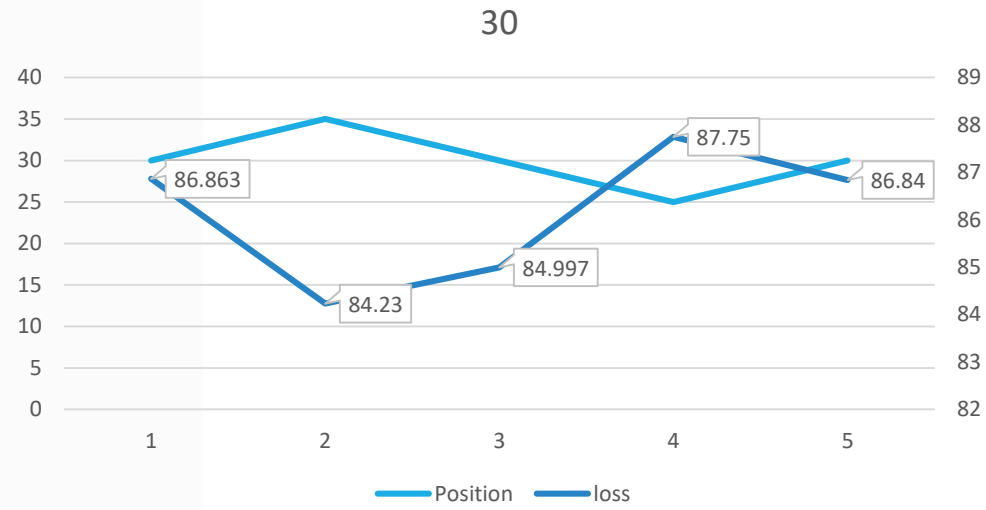


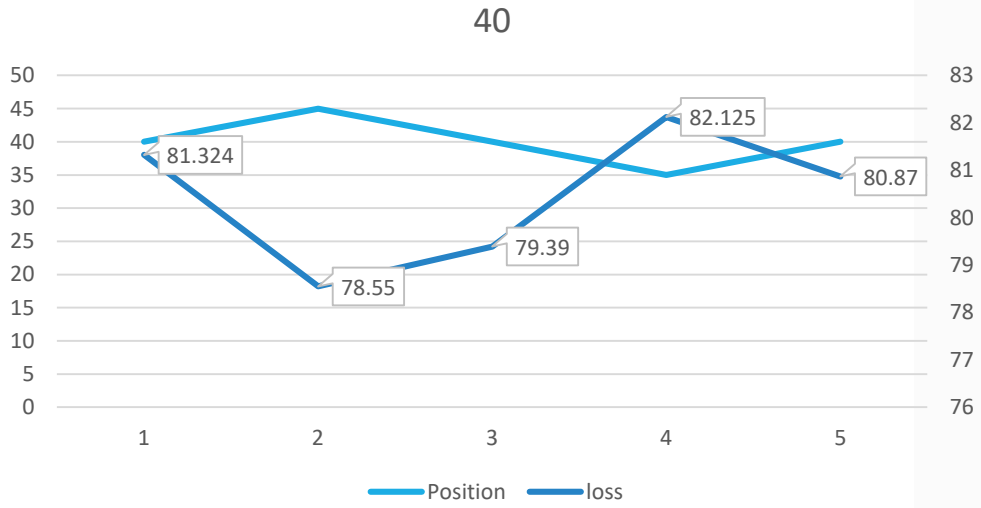
Motor Analysis

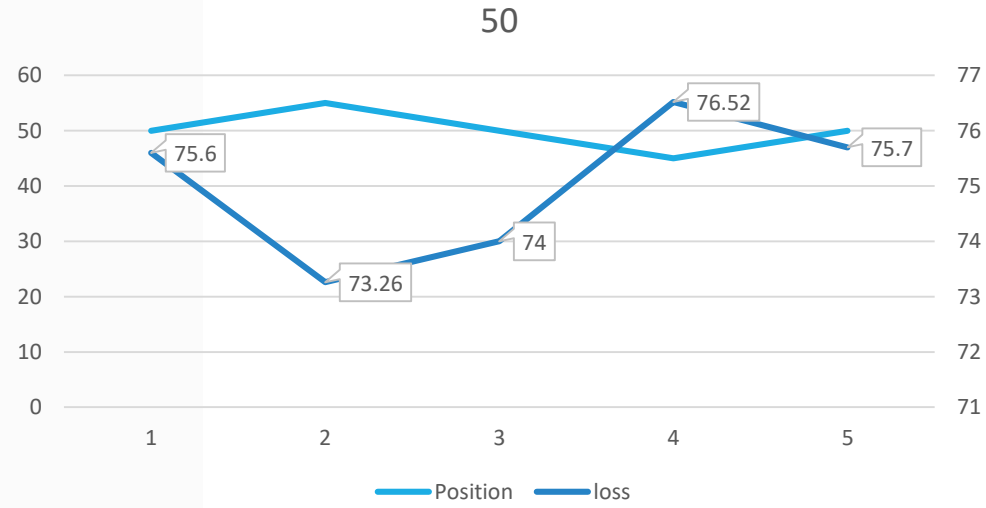
- Conducted tests to analyze motor behavior under different conditions Varied key parameters.
- Performed a dedicated study to determine motor backlash.

Analysis Outcomes

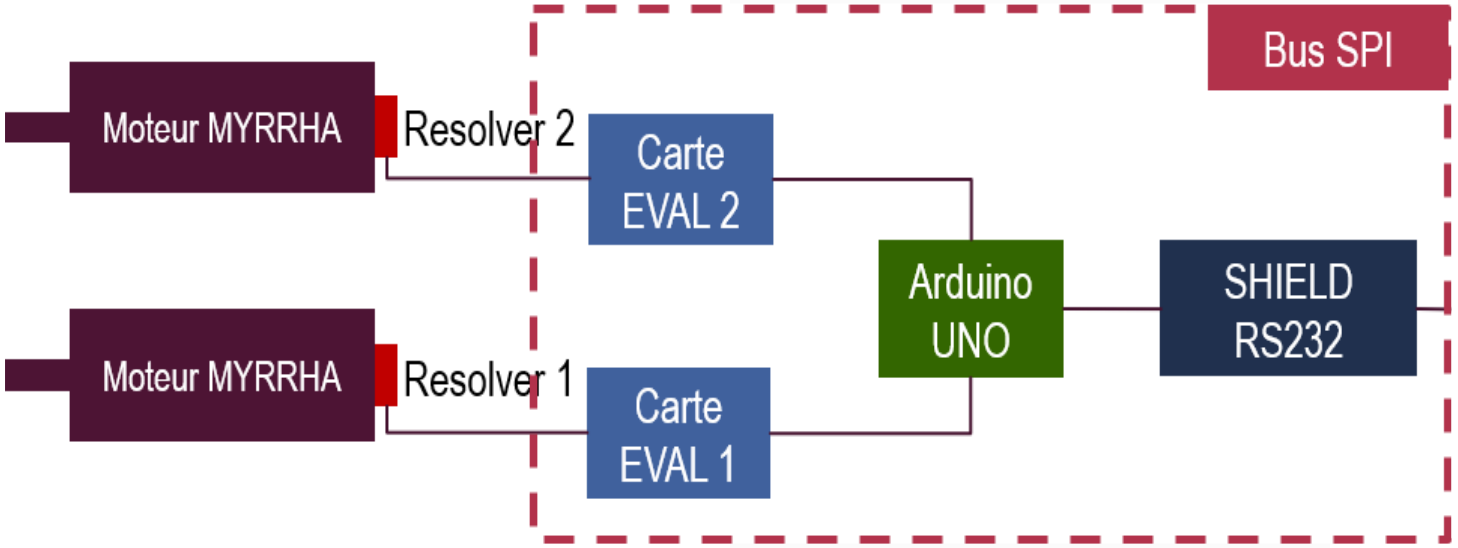








Current Development Workflow

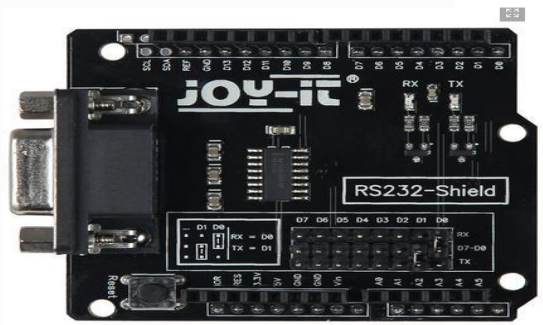


System Components and Instruments

Arduino UNO

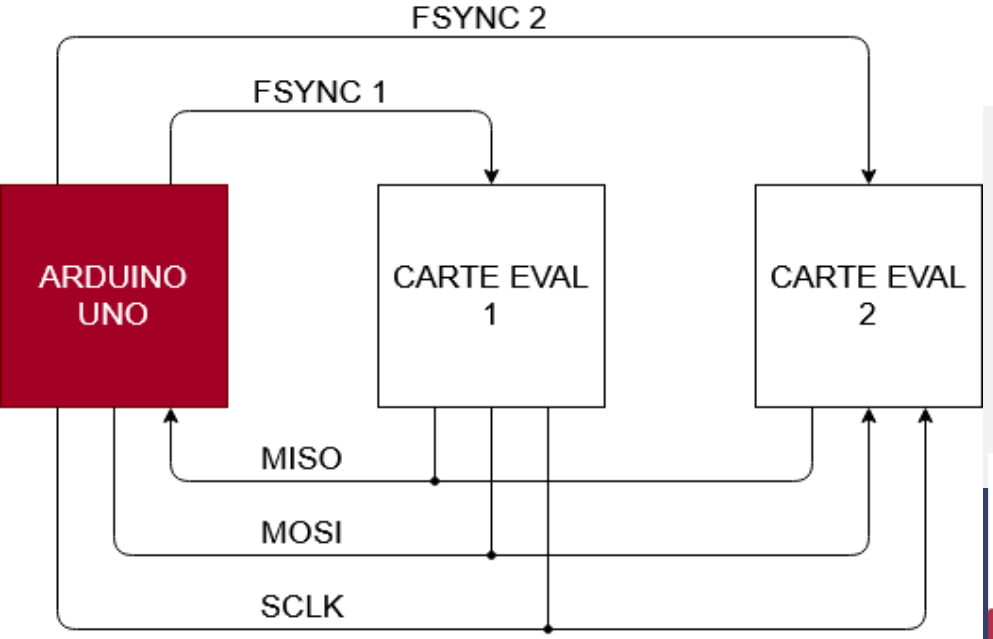
Stepper Motor

EVAL-CN273-SDPZ

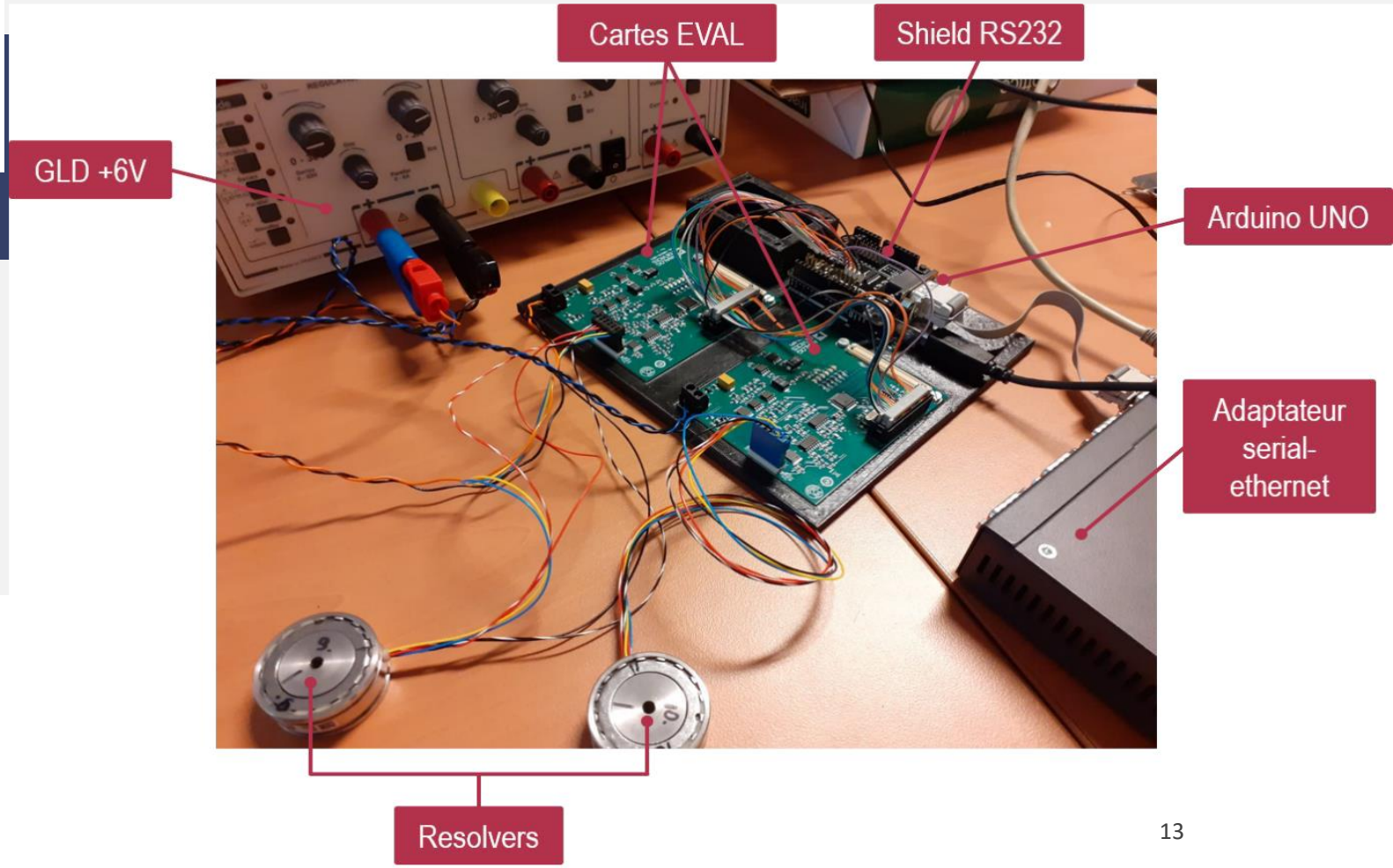


Shield RS232

System integration



Integrated two stepper motors into the system for synchronized movement Used CN0276 with AD2S1210 to read: Position Speed Error conditions Enabled bidirectional communication between Arduino and the sensor chip using SPI protocol.



System Programing Code

```ino
#include<SPI.h>

// pins
const int PINA0 = 3; // select data
const int PINA1 = 2; // select data
const int Sample = 7; // update data
const int SOE = 6; // always Low 0
const int RES0 = 4; // resolution setting
const int RES1 = 5; // resolution setting
const int CS1 = 10; // chip select 1
const int CS2 = 8; // chip select 2

void setNormalModePosition() {

  digitalWrite(PINA0,LOW);
  digitalWrite(PINA1,LOW);
  delayMicroseconds(20);

}

void setNormalModeVelocity() {

  digitalWrite(PINA0,LOW);
  digitalWrite(PINA1,HIGH);
  delayMicroseconds(20);



}

void setConfigurationMode() {

  digitalWrite(PINA0,HIGH);
  digitalWrite(PINA1,HIGH);
  delayMicroseconds(25);

}

void setResolution16Bit(){

  digitalWrite(RES0,HIGH);
  digitalWrite(RES1,HIGH);

}

void sampleData(){

  digitalWrite(Sample,HIGH);
  delayMicroseconds(1);
  digitalWrite(Sample,LOW);
  delayMicroseconds(1);
  digitalWrite(Sample,HIGH);
  delayMicroseconds(1);
  digitalWrite(Sample,LOW);
  delayMicroseconds(1);

}


void setup() {

  Serial.begin(115200);

  // Pin mode
  pinMode(PINA0,OUTPUT);
  pinMode(PINA1,OUTPUT);
  pinMode(Sample,OUTPUT);
  pinMode(SOE,OUTPUT);
  pinMode(RES0,OUTPUT);
  pinMode(RES1,OUTPUT);
  pinMode(CS1,OUTPUT);
  pinMode(CS2,OUTPUT);

  digitalWrite(SOE,LOW);
  digitalWrite(CS1,HIGH);
  digitalWrite(CS2,HIGH);

  setResolution16Bit();

  //

  SPI.begin();
  SPI.beginTransaction(SPISettings(1000000,MSBFIRST,SPI_MODE1));
  delay(100);

}

float readPosition1(){

  // uint16_t result = 0;
  setNormalModePosition();
  sampleData();
  digitalWrite(CS2,HIGH);
  digitalWrite(CS1,LOW);
  delayMicroseconds(1);

  uint16_t msb1 = SPI.transfer(0x00);
  uint16_t lsb1 = SPI.transfer(0x00);
  uint16_t RAW1 = (msb1<<8) | lsb1;
  float result = ((float)RAW1*360.0)/65536.0;

  digitalWrite(CS1,HIGH);

  return result;

}
```

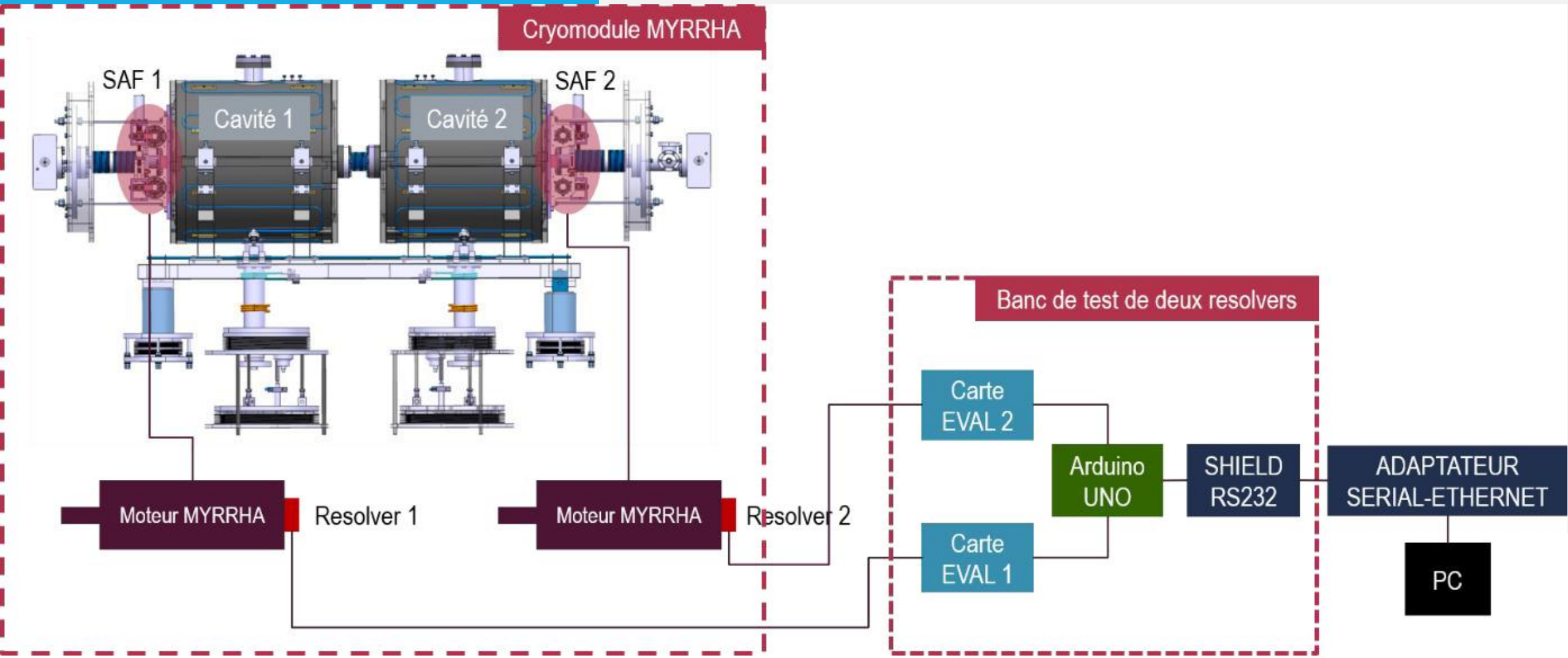
```
106  }
107
108  float readVelocity1(){
109
110    //int16_t result = 0;
111    setNormalModeVelocity();
112    sampleData();
113    digitalWrite(CS2,HIGH);
114    digitalWrite(CS1,LOW);
115    delayMicroseconds(1);
116
117    uint8_t msbV = SPI.transfer(0x00);
118    uint8_t lsbV = SPI.transfer(0x00);
119    int16_t RAWV = ((int16_t)msbV<<8) | lsbV;
120    float resultV =  0.004*(float)RAWV;
121
122    digitalWrite(CS1,HIGH);
123
124    return resultV;
125  }
126
127  float readPosition2(){···
144  }
145
146  float readVelocity2(){···
164  }
165
166  uint8_t readConfigurationResisterC1(uint8_t regAddress){
167    sampleData();
168    delayMicroseconds(5);
169    setConfigurationMode();
170
171    digitalWrite(CS2,HIGH);
172    digitalWrite(CS1,LOW);
173    delayMicroseconds(1);
```

```
172    digitalWrite(CS1,LOW);
173    delayMicroseconds(1);
174    SPI.transfer(regAddress);
175    delayMicroseconds(5);
176
177    digitalWrite(CS1,HIGH);
178    delayMicroseconds(5);
179    digitalWrite(CS1,LOW);
180    delayMicroseconds(1);
181
182    uint8_t read = SPI.transfer(regAddress);
183    delayMicroseconds(1);
184    digitalWrite(CS1,HIGH);
185
186    return read;
187
188  }
189
190  void readStatusC1(){
191
192    uint8_t status = readConfigurationResisterC1(0x92);
193    Serial.print("status of C1: 0x");
194    Serial.println(status, HEX);
195
196  }
197
198  void writeConfigRegisterC1(uint8_t regAddress, uint8_t value) {
199    setConfigurationMode();
200    digitalWrite(CS2,HIGH);
201    digitalWrite(CS1, LOW);
202    SPI.transfer(regAddress);
203    delayMicroseconds(5);
204
205    digitalWrite(CS1,HIGH);
206    delayMicroseconds(5);
```

```
209
210   SPI.transfer(value);
211   digitalWrite(CS1, HIGH);
212 }
213
214 void configMenu() {
215   Serial.println("     CONFIG MENU     ");
216   Serial.println("1. Write to register Excitation Frequency 0x91");
217   Serial.println("2. Write to register Control 0x92");
218   Serial.println("3. Write to register Soft Reset  0xF0");
219
220
221   while (!Serial.available());
222   String choiceStr = Serial.readStringUntil('\n');
223   choiceStr.trim();
224
225   int choice = choiceStr.toInt();
226
227   uint8_t regAdd;
228   if (choice == 1) regAdd = 0x91;
229   else if (choice == 2) regAdd = 0x92;
230   else if (choice == 3) regAdd = 0xF0;
231   else {
232     Serial.println("Invalid choice");
233     return;
234   }
235
236   Serial.println("Enter value to write in HEX:");
237   while (!Serial.available());
238   String DATA = Serial.readStringUntil('\n');
239   DATA.trim();
240   uint8_t val = (uint8_t) strtol(DATA.c_str(), NULL, 16);
241
242   writeConfigRegisterC1(regAdd, val);
243
```

```
246 void loop() {
247
248   float Velocity1 = readVelocity1();
249   Serial.print("Velocity1: ");
250   Serial.print(Velocity1);
251   Serial.println(" rps");
252   //delay(1500);
253
254   float position1 = readPosition1();
255   Serial.print("position1: ");
256   Serial.print(position1);
257   Serial.println(" Deg");
258
259   delay(500);
260   //delayMicroseconds(100);
261
262   float Velocity2 = readVelocity2();
263   Serial.print("Velocity2: ");
264   Serial.print(Velocity2);
265   Serial.println(" rps");
266   //delay(1500);
267
268   float position2 = readPosition2();
269   Serial.print("position2: ");
270   Serial.print(position2);
271   Serial.println(" Deg ");
272   delay(1000);
273
274   configMenu();
275   delay(2000);
276
277   readStatusC1 ();
278   delay(2000);
279
280 }
```

Final system Layout



Thank you

QUESTIONS?