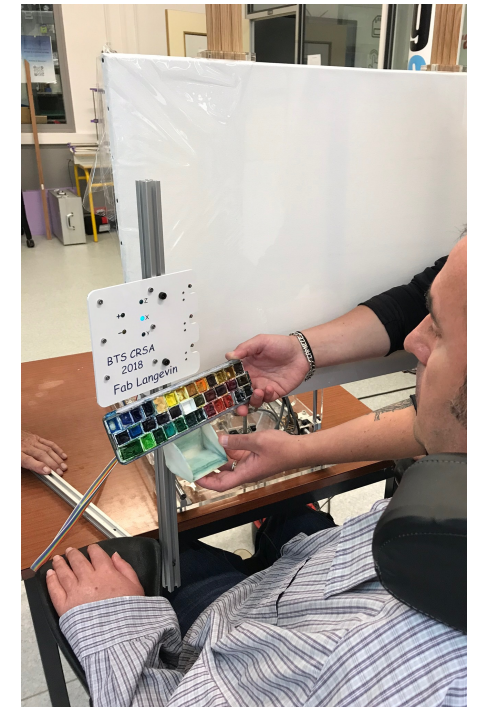
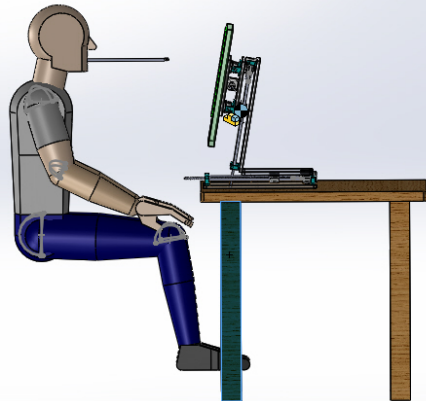
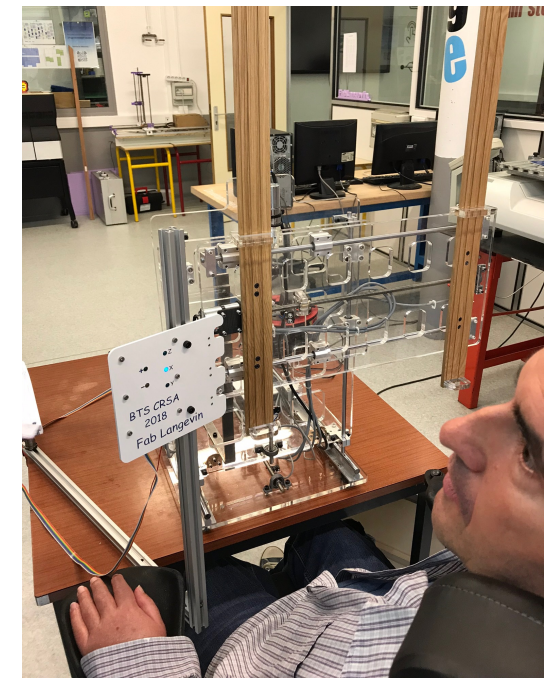
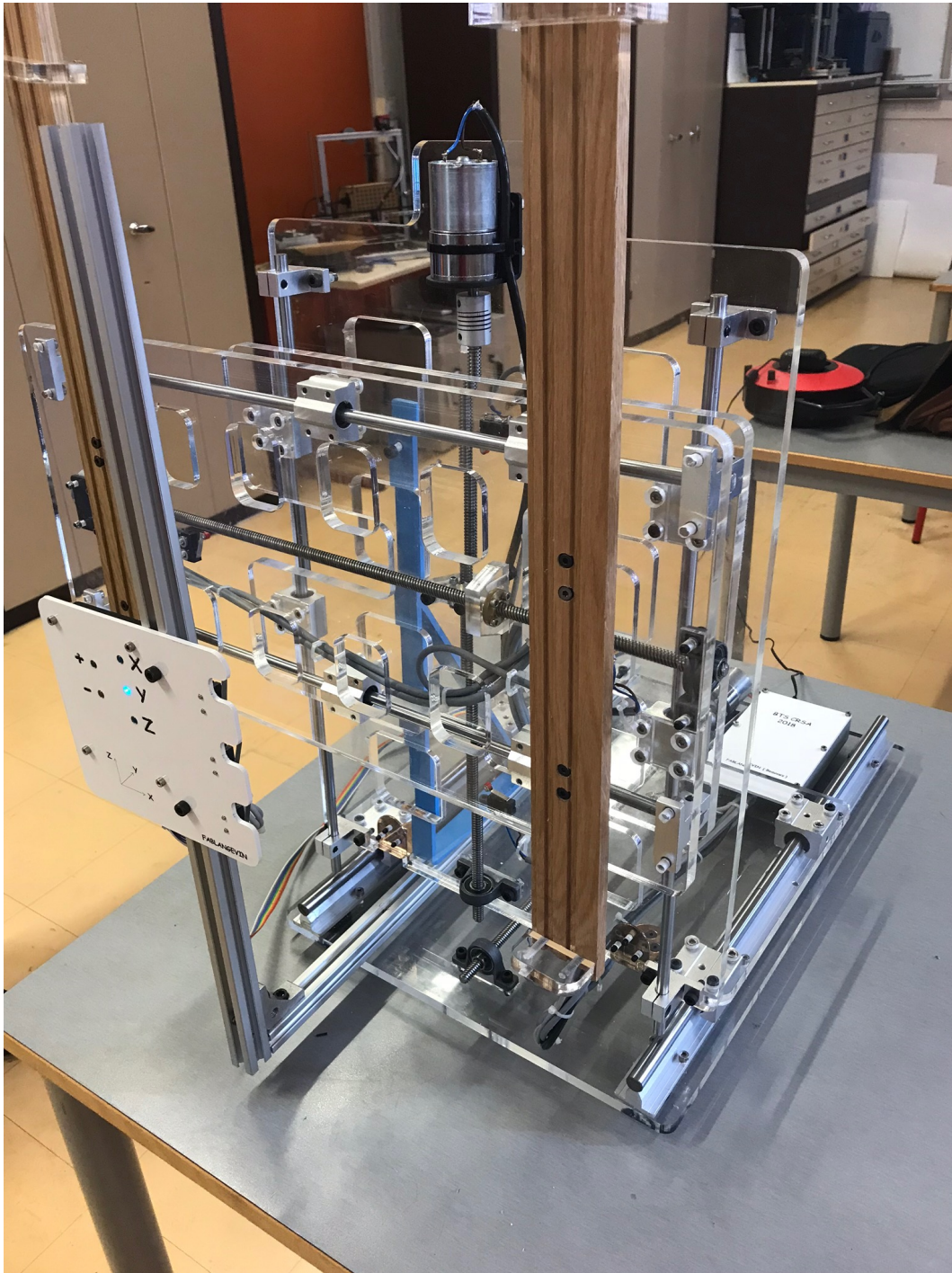
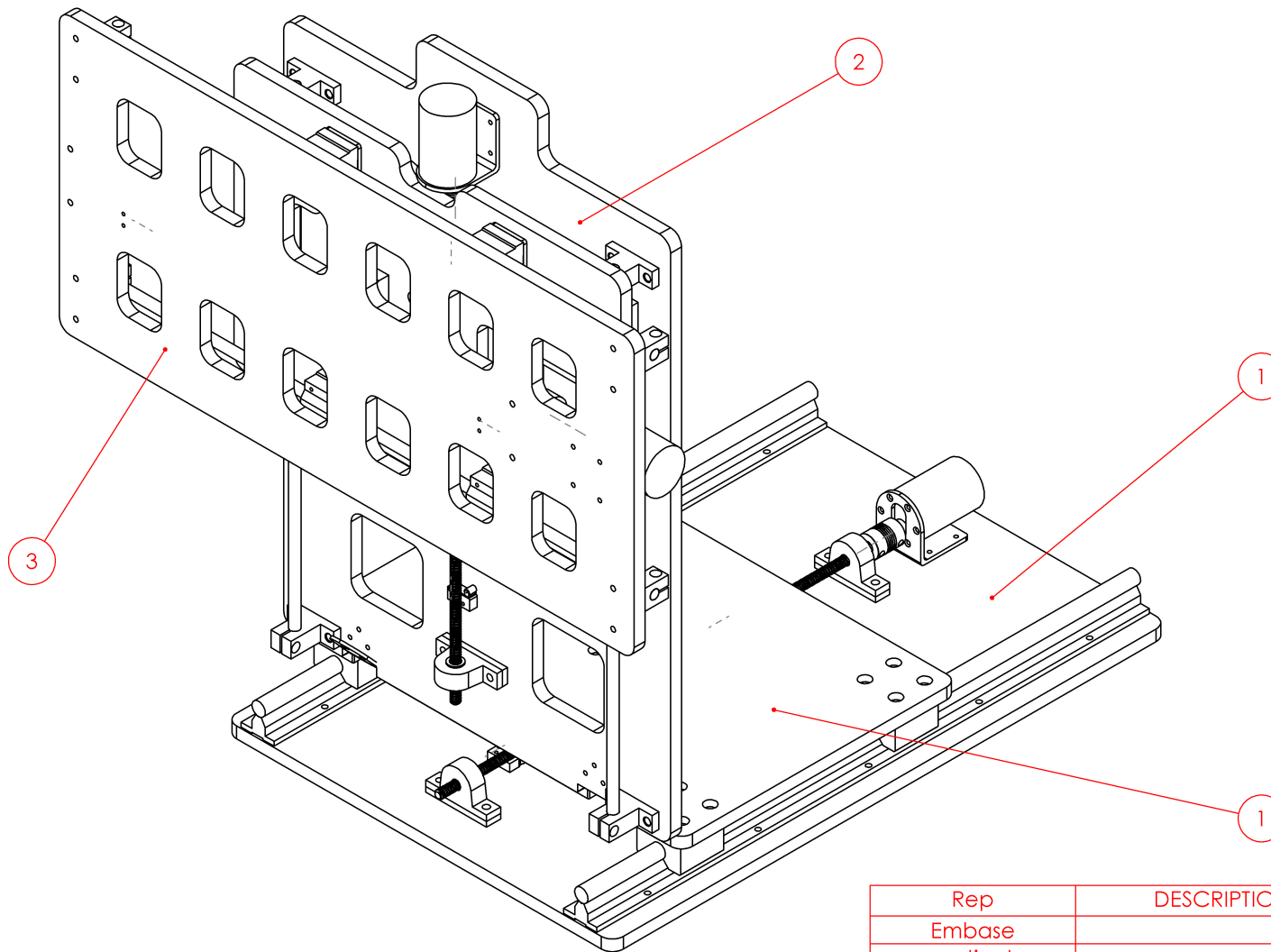


Chevalet motorisé



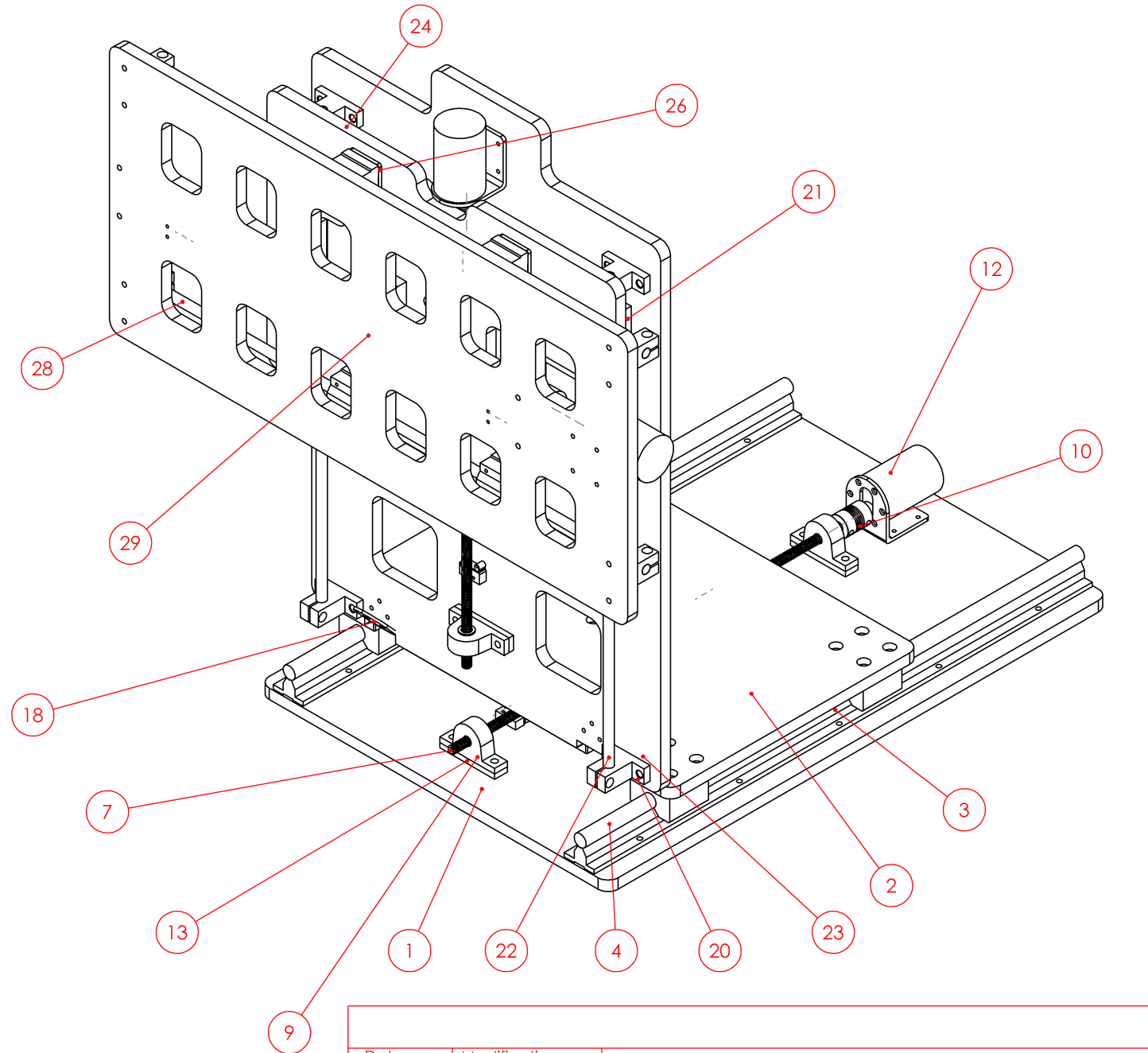




Rep	DESCRIPTION	Matière	rep	Commentaires
Embase			1	
vertical			2	
Horizontal			3	
boitier led			4	

Dates		Modifications		Chevalet motorisé			
Nom fichier SW:		chevalet motorisé version 2					
Echelle : 1/3	Format : A3	Date : 02/01/2018	ATEL SECT	MACHINE/PROJET	N° MOULE / ENS	N° PLAN	
Dessiné par :		St Bouillette		ensemble			
Contrôlé par :							

No. ARTICLE	NUMERO DE PIECE	QTE
1	plaque embase	1
2	plaque basse	1
3	support tige 12 mm	2
4	tige de 12mm	2
5	coulisseau 12 mm	4
6	support vis	1
7	VIS embase	2
8	Colleretes	3
9	palier semelle	6
10	Accouplement	3
11	equerre	3
12	moteur	3
13	cale	6
14	Bras levier	9
15	Bras levier rouleau interrupteur	9
16	Rouleau	9
17	Tube fixe rouleau	9
18	attache charniere	2
19	millieu charniere	1
20	support tige 8 mm	8
21	SCE16UU	8
22	tige droite verticale	2
23	plaque verticale	1
24	plaque intermediaire	2
25	support vis vertical	2
26	cale douille	8
27	dbl_815x	4
28	tige droite horizontale	2
29	plaque horizontale	1
30	vis horizontal	1
31	boitier led	1
32	breadboard	1
33	joue boitier led	1
34	support led	1



Dates	Modifications					
		Chevalet motorisé				
Nom fichier SW:						
chevalet motorisé version 2						
Echelle : 1/3	Format : A3	Date : 02/01/2018	ATEL/SECT	MACHINE/PROJET	N° MOULE / ENS	N° PLAN
		Dessiné par :	St Bouillette		ensemble(2)	
		Contrôlé par :				

Programme arduino :

```
#include <AFMotor.h>

// moteur CC branché sur M1 M2 et M3
AF_DCMotor motorX(1);
AF_DCMotor motorY(2);
AF_DCMotor motorZ(3);

// déclaration des pins

const int ledXpin = 32; // allume la led du moteur 1
const int ledYpin = 34; // allume la led du moteur 2
const int ledZpin = 36; // allume la led du moteur 3
const int ledtestantitrigopin = 43; // allume la led test bouton 2
const int ledtesttrigopin = 45; // allume la led test bouton 3

const int BoutonpinXYZ = 24; // entree pour lire la valeur des boutons
const int Boutonpintrigo = 25;
const int Boutonpinantitrigo = 26;
const int Findecourse1 = 50;
const int Findecourse2 = 52;

// valeur de base des variables

int moteurselection = 0;
int valeurBouton = 0;
int numeroBouton = 0;
bool relachement = false;

void setup()

    pinMode(ledtestantitrigopin, OUTPUT);
    pinMode(ledtesttrigopin, OUTPUT);
    pinMode(ledXpin, OUTPUT);
    pinMode(ledYpin, OUTPUT);
    pinMode(ledZpin, OUTPUT);

    digitalWrite(ledZpin, HIGH);

    // Configuration des boutons poussoirs

    pinMode(BoutonpinXYZ, INPUT_PULLUP);
    pinMode(Boutonpintrigo, INPUT_PULLUP);
    pinMode(Boutonpinantitrigo, INPUT_PULLUP);
    pinMode(Findecourse1, INPUT_PULLUP);
    pinMode(Findecourse2, INPUT_PULLUP);
}

void loop() {

    delay (50);

    if ((digitalRead(BoutonpinXYZ) == LOW) && (!relachement) ) // le BP est le n°3 ( à gauche )
        {
            moteurselection = (moteurselection + 1) % 3 ;
            // on augmente la valeur de motoselect en appliquant un modulo
            relachement = true ;

        }

    else if (digitalRead(BoutonpinXYZ) == HIGH )
        {
            relachement = false ;
            numeroBouton = 0 ;
            directionmotor(numeroBouton) ;
            digitalWrite(ledtesttrigopin, LOW);
            digitalWrite(ledtestantitrigopin, LOW);
        }

    if (digitalRead(Boutonpintrigo) == LOW && digitalRead(Findecourse1) == HIGH) // le BP est le n°1 ( à droite)
        {

            numeroBouton = 1;
            directionmotor(numeroBouton) ;
            digitalWrite(ledtesttrigopin, HIGH);
        }

    if (digitalRead(Boutonpinantitrigo) == LOW && digitalRead(Findecourse2) == HIGH) // le BP est le n°2 ( au centre )
        {
            numeroBouton = 2;
            directionmotor(numeroBouton) ;
            digitalWrite(ledtestantitrigopin, HIGH);
        }

        rotationMoteur(moteurselection);
        ledAxis(moteurselection) ;
    }

    void directionmotor(int numeroBouton)

        {
            switch(numeroBouton)
            {
            case 0:
                motorX.run(RELEASE);
                motorY.run(RELEASE);
                motorZ.run(RELEASE);
                break;

            case 1:
                motorX.run(BACKWARD);
                motorY.run(BACKWARD);
                motorZ.run(BACKWARD);
                break;

            case 2:
                motorX.run(FORWARD);
                motorY.run(FORWARD);
            }
        }
}
```

```

    motorZ.run(FORWARD);
    break;
}
}

```

```
void rotationMoteur(int motorselection)
```

```

{
/* la variable direction prend 3 valeurs
* 0 : rotation moteur X
* 1 : rotation moteur Y
* 2 : rotation moteur Z
*/

```

```
switch(motorselection)
```

```

{
case 0:

    motorX.setSpeed(255);
    motorY.setSpeed(0);
    motorZ.setSpeed(0);
    break;

```

```
case 1:
```

```

    motorX.setSpeed(0);
    motorY.setSpeed(255);
    motorZ.setSpeed(0);
    break;

```

```
case 2:
```

```

    motorX.setSpeed(0);
    motorY.setSpeed(0);
    motorZ.setSpeed(255);
    break;

```

```

}
}

```

```
void ledAxis(int motorselection)
```

```

{
/* la variable direction prend 3 valeurs
* 0 : allume la led verte
* 1 : allume la led jaune
* 2 : allume la led bleu
*/

```

```
switch(motorselection)
```

```

{
case 0:
    digitalWrite(ledXpin,HIGH);
    digitalWrite(ledYpin,LOW);
    digitalWrite(ledZpin,LOW);
    break;

```

```

case 1:
    digitalWrite(ledXpin,LOW);
    digitalWrite(ledYpin,HIGH);
    digitalWrite(ledZpin,LOW);
    break;

```

```

case 2:
    digitalWrite(ledXpin,LOW);
    digitalWrite(ledYpin,LOW);
    digitalWrite(ledZpin,HIGH);
    break;

```

```

}

```

```

}

```