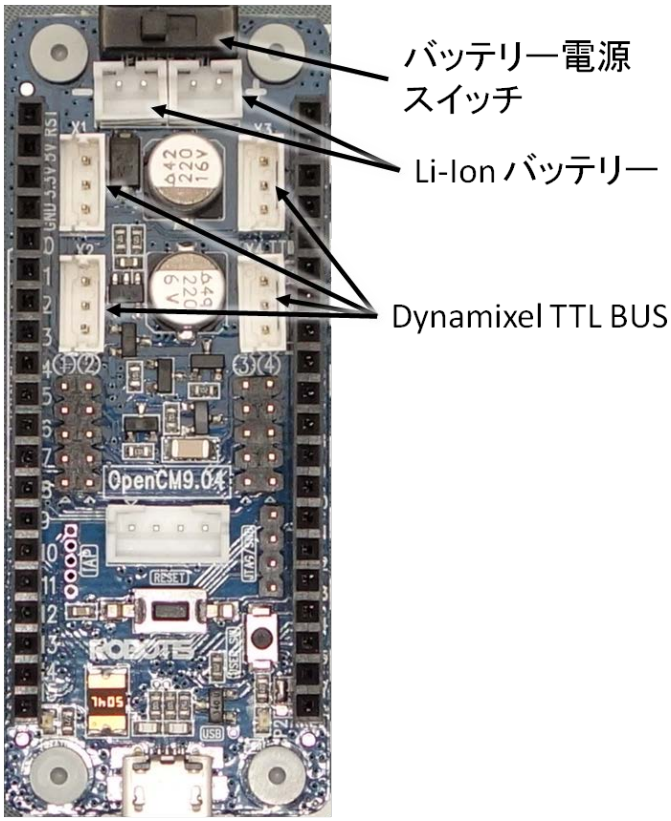
11. モータを回してみよう

LED や 7 セグメントを使って，光を出したり文字を出力したりするのは，マイコンを用いて情報の世界と実世界を繋ぐことに該当する．しかし視覚的な表現だけでは，実世界でできることには限りがある．やはり物理的にモノを動かしたり，サービスを行ったりすることで，マイコンを使用するメリットがもっと実感できる．モータの種類には，DC モータ，ステップモータ，超音波モータ，など様々な種類があるが，ここではデジタルサーボモータを用いる．普通のモータを使用する場合には，モータ以外に様々な周辺回路とモータ制御のための複雑なプログラミングが必要となる．しかし，デジタルサーボモータには，モータ内に周辺電子回路が用意されており，さらに小型のマイコンが入っており，その中にモータ制御のためのプログラムも既に用意されている．そのため，マイコンからデジタルサーボモータに命令を転送するだけで，指示通りに容易にモータを動かすことができる．電気電子回路や制御理論などについて十分な知識がない人にとって，デジタルサーボモータは非常に便利で役立つものである．

ここで使用するデジタルサーボモータは XL-320 というモデルで，OpenCM に直接繋ぐことが可能であり，簡単に使える．まずは専用のケーブルを用いて XL-320 1 台と OpenCM を繋ごう．XL-320 の動作電圧は 6.0 ～ 8.4V であるため，USB ケーブルだけを接続して PC から OpenCM に電力を供給した場合には電圧が低く，XL-320 は正しく動かなかったり，動いたとしてもトルクが非常に弱い．したがって，XL-320 を利用する場合には，OpenCM に DC アダプタを接続するか，バッテリを接続して動作させよう．ここでは専用のバッテリーを接続することとする．以下の図を参考に OpenCM の正しい場所にバッテリー(Li-Ion バッテリー)と XL-320（Dynamixel TTL BUS）を接続しよう．バッテリーを接続した場合には，OpenCM のパ電源スイッチを用いてバッテリーの電源を On/Off することが可能である．



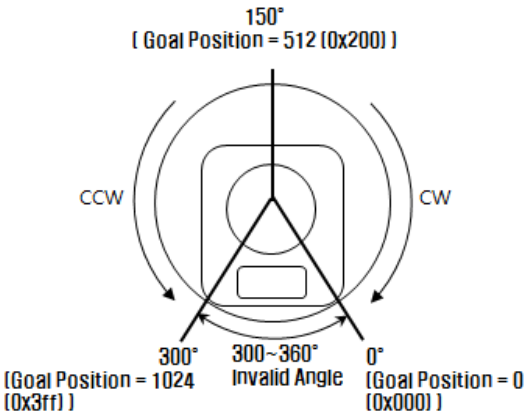
11.1 モータを動かしてみよう

[ファイル]-[スケッチの例]-[07.Dynamixel Easy]から i_goalPosition を選択して開いてみよう.

```
#define ID_NUM 1

Dynamixel Dxl(DXL_BUS_SERIAL1);

void setup() {
  // Initialize the dynamixel bus:
  // Dynamixel 2.0 Baudrate -> 0: 9600, 1: 57600, 2: 115200, 3: 1Mbps
  Dxl.begin(3);
  Dxl.jointMode(ID_NUM); //jointMode() is to use position mode
}

void loop() {
  Dxl.goalPosition(ID_NUM, 1); //ID 1 dynamixel moves to position 1
  delay(2000);
  Dxl.goalPosition(ID_NUM, 1023);//ID 1 dynamixel moves to position 1023
  delay(2000);
}
```

XL-320 は 1 台のマイコンに同時に 253 台まで繋ぐことが可能である. それぞれの XL-320 は ID 番号を持っており, 2 台以上の XL-320 が繋ぐ場合は, 必ず互いに異なる ID 番号にする必要がある. 初期状態の XL-320 の ID 番号は 1 である. 上記のプログラムでは, XL-320 が初期状態であることを前提にしているため, #define ID_NUM 1 で ID_NUM を 1 として定義している. もし ID 番号が 1 以外の XL-320 を使用する場合は, 1 の代わりに使用する XL-320 の ID 番号にするか, XL-320 の ID番号を 1 に変えておく必要がある. XL-320 の ID 番号を変更する方法は後述する.

Dynamixel Dxl( );は OpenCM で XL-320 を使用するときに必要な命令で, DXL_BUS_SERIAL1 の値は 1 である. Dxl.begin(3); は XL-320 との通信速度を設定する命令で, 初期状態の XL-320 は 1Mbps に設定されているため, それに合わせて引数を 3 にしている. Dxl.jointMode(ID_NUM);は, ID_NUM の ID 番号を持っている XL-320 を関節モードで使用することを宣言する命令である. XL-320 には, 人の関節のように使用できる関節モードと車の車輪のように無限回転する車輪モードがある. Dxl.goalPosition(ID_NUM, **位置**);は ID_NUM の XL-320 を**位置**まで移動させる命令である. **位置**とは正しく言うと角度のことで, 値と実際の角度間の関係は, 右図を参考にせよ.



11.2 回転速度を変えてみよう

次に, [ファイル]-[スケッチの例]-[07.Dynamixel Easy]から j_goalSpeed を選択して開いてみよう.

```
#define ID_NUM 1

Dynamixel Dxl(DXL_BUS_SERIAL1);

void setup() {
  // Initialize the dynamixel bus:
  // Dynamixel 2.0 Baudrate -> 0: 9600, 1: 57600, 2: 115200, 3: 1Mbps
  Dxl.begin(3);

  Dxl.goalSpeed(ID_NUM, 100);  //Dynamixel ID 1 Speed Control 100 setting
  Dxl.jointMode(ID_NUM); //jointMode() is to use position mode
}

void loop() {
  Dxl.goalPosition(ID_NUM, 1); //ID 1 dynamixel moves to position 1
  delay(1000);
  Dxl.goalPosition(ID_NUM, 300);//ID 1 dynamixel moves to position 300
  delay(1000);
}
```

このプログラムは i_goalPosition とほとんど同じであるが, setup( )内で Dxl.goalSpeed(ID_NUM, 100);を追加している点と動作間の delay 時間を短くしている部分が異なる. Dxl.goalSpeed(ID_NUM, **速度**);は ID_NUM の XL-320 の回転速度を**速度**に設定する命令である. **速度**の範囲は 0～1023 である.

loop()内で Dxl.goalSpeed(ID_NUM, **速度**);を使って, それぞれの動作ごとに異なる速度で回転するようにしてみよう.

11.3 車輪モードで動かしてみよう

[ファイル]-[スケッチの例]-[07.Dynamixel Easy]から d_wheelMode を選択して開いてみよう.

```
void setup() {
  // Initialize the dynamixel bus:

  // Dynamixel 2.0 Baudrate -> 0: 9600, 1: 57600, 2: 115200, 3: 1Mbps
  Dxl.begin(3);
  Dxl.wheelMode(ID_NUM); //wheelMode() is to use wheel mode
}

void loop() {
  Dxl.goalSpeed(ID_NUM, 400); //forward
```

```
  delay(5000);
  Dxl.goalSpeed(ID_NUM, 400 | 0x400); //reverse
  delay(5000);
  Dxl.goalSpeed(ID_NUM, 0); //stop
  delay(2000);
}
```

これは XL-320 を車輪モードで使用する例である. setup()内で Dxl.wheelMode(ID_NUM);が宣言されており, ID 番号が ID_NUM の XL-320 を車輪モードに切り替えている. このモードにすると速度を 0 以外に設定すると, 時計回りまたは半時計周りの回転をする. Dxl.goalSpeed(ID_NUM, **速度**); は ID_NUM の ID 番号をもつ XL-320 を**速度**で回転させる命令であるが, ここでは車輪モードに設定されているため, Dxl.goalPosition(ID_NUM, **位置**); の命令がなくても回転を始める. 速度の範囲は 0～1023 または 1024-2047 で, 前者は XL-320 を反時計回りに, 後者は時計回りに回転させる. Dxl.goalSpeed(ID_NUM, 400 | 0x400);は 400 に 0x400 を OR して, すなわち**速度**を 1424 にして, XL-320 を時計回りに速度 400 で回転させるという意味である. 0x は 16 進数であることを意味し, 0x400 は 1024 を意味する. 1023 以下の数字と 0x400 を | (OR 演算子)することは 1024 を加算するのと同じ意味である.

11.4 ID を設定しよう

[ファイル]-[スケッチの例]-[07.Dynamixel Easy]から b_setID を選択して開いてみよう.

```
#define NEW_ID 2

Dynamixel Dxl(DXL_BUS_SERIAL1);

void setup() {
  // Initialize the dynamixel bus:
  // Dynamixel 2.0 Baudrate -> 0: 9600, 1: 57600, 2: 115200, 3: 1Mbps
  Dxl.begin(3);
/*************** CAUTION ************************
 * All dynamixels in bus will be changed to ID 2 by using broadcast ID(0xFE)
 * Please check if there is only one dynamixel that you want to change ID
 ************************************************/
  Dxl.setID(BROADCAST_ID, NEW_ID);  //Dynamixel_Id_Change 1 to 2
  Dxl.jointMode(NEW_ID); //jointMode() is to use position mode
}

void loop() {
  /*Turn dynamixel ID 2 to position 0*/
  Dxl.goalPosition(NEW_ID, 0);
  // Wait for 1 second (1000 milliseconds)
  delay(1000);
  /*Turn dynamixel ID 2 to position 300*/
```

```
  Dxl.goalPosition(NEW_ID, 300);
  // Wait for 1 second (1000 milliseconds)
  delay(1000);
}
```

このプログラムは，Dxl.setID(BROADCAST_ID, NEW_ID);命令を用いて OpenCM に繋がっている全ての XL-320 の ID 番号を NEW_ID に設定している．このプログラムでは，NEW_ID が 2 と宣言されているため，XL-320 の ID 番号は 2 番になる．BROADCAST_ID は 254 と定義されており，この番号を使用すると全ての XL-320 が Dxl.setID()命令の対象になる．

ここで注意すべき点は，Dxl.setID(BROADCAST_ID, NEW_ID);をすると，OpenCM に繋がっている全ての XL-320 の ID 番号が NEW_ID に設定されることである．そのため，必ず XL-320 を 1 台だけを繋いで ID を設定しなければいけない．また，OpenCM では電源を入れた瞬間に最後にダウンロードされたプログラムが実行されるため，ID を変更するプログラムをコンパイル・ダウンロード後に ID を変更したい XL-320 を接続しよう．

上記のプログラムを用いて 1 台の XL-320 の ID を 2 番に変更しよう．その後，ID1 の XL-320 と ID2 の XL-320 を OpenCM に接続し，XL-320 それぞれに異なる指令を出して異なる動きをさせてみよう．（注意：XL-320 を繋ぐ前に ID 番号変更プログラムを他のプログラムに書き換えておこう．そうしないと，接続した 2 台の XL-320 の ID が全て同じ ID に変更されてしまう．）

11.5 XL-320 の参考資料

以下の表は XL-320 を参考資料である. 必要に応じて参考にしよう.

領域	アドレス (16進数)	サイズ (byte)	名称	説明	アクセス	初期値	最小値	最大値
EEPROM	0	2	Model Number	Model number	R	350	-	-
	2	1	Version of Firmware	Information on the version of firmware	R	-	-	-
	3	1	ID	ID of Dynamixel	RW	1	0	252
	4	1	Baud Rate	Baud Rate of Dynamixel	RW	3	0	3
	5	1	Return Delay Time	Return Delay Time	RW	250	0	254
	6	2	CW Angle Limit	clockwise Angle Limit	RW	0	0	1023
	8	2	CCW Angle Limit	counterclockwise Angle Limit	RW	1023	0	1023
	11	1	Control Mode	Control Mode	RW	2	1	2
	12	1	Limit Temperature	Internal Limit Temperature	RW	65	0	150
	13	1	lower Limit Voltage	Lowest Limit Voltage	RW	60	50	250
	14	1	Upper Limit Voltage	Upper Limit Voltage	RW	90	50	250
	15	2	Max Torque	Lowest byte of Max. Torque	RW	1023	0	1023
	17	1	Return Level	Return Level	RW	2	0	2
	18	1	Alarm Shutdown	Shutdown for Alarm	RW	3	0	7
RAM	24	1	Torque Enable	Torque On/Off	RW	0	0	1
	25	1	LED	LED On/Off	RW	0	0	7
	27	1	D Gain	D Gain	RW	0	0	254
	28	1	I Gain	I Gain	RW	0	0	254
	29	1	P Gain	P Gain	RW	32	0	254
	30	2	Goal Position	Goal Position	RW	-	0	1023
	32	2	Goal Velocity	Goal Speed	RW	-	0	2047
	35	2	Goal Torque	Goal Torque	RW	-	0	1023
	37	2	Present Position	Current Position	R	-	-	-
	39	2	Present Speed	Current Speed	R	-	-	-
	41	2	Present Load	Current Load	R	-	-	-
	45	1	Present Voltage	Current Voltage	R	-	-	-
	46	1	Present Temperature	Present temperature	R	-	-	-
	47	1	Registered Instruction	Registered Instruction	R	0	-	-
	49	1	Moving	Moving	R	0	-	-
	50	1	Hardware Error Status	Hardware error status	R	0	-	-
	51	2	Punch	Punch	RW	32	0	1023