Stepper Motors

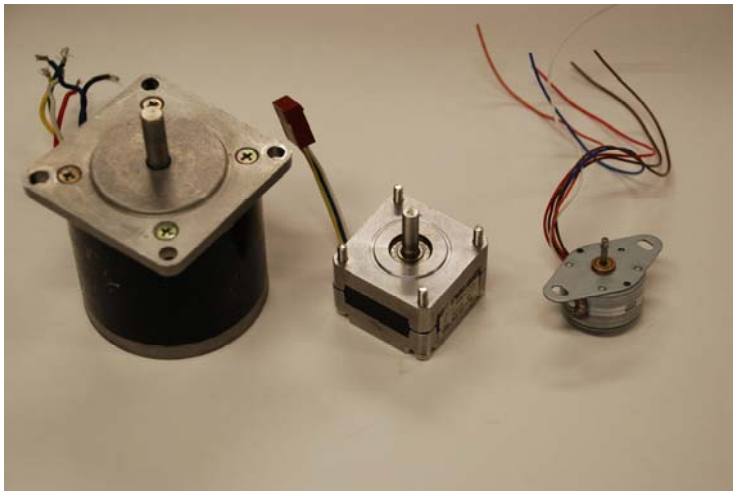


Figure 8.24. Photo of three stepper motors.

http://users.ece.utexas.edu/~valvano/Datasheets/

Look for data sheets with Stepper in it.

Comes in 24,48,200,400 steps/rotation

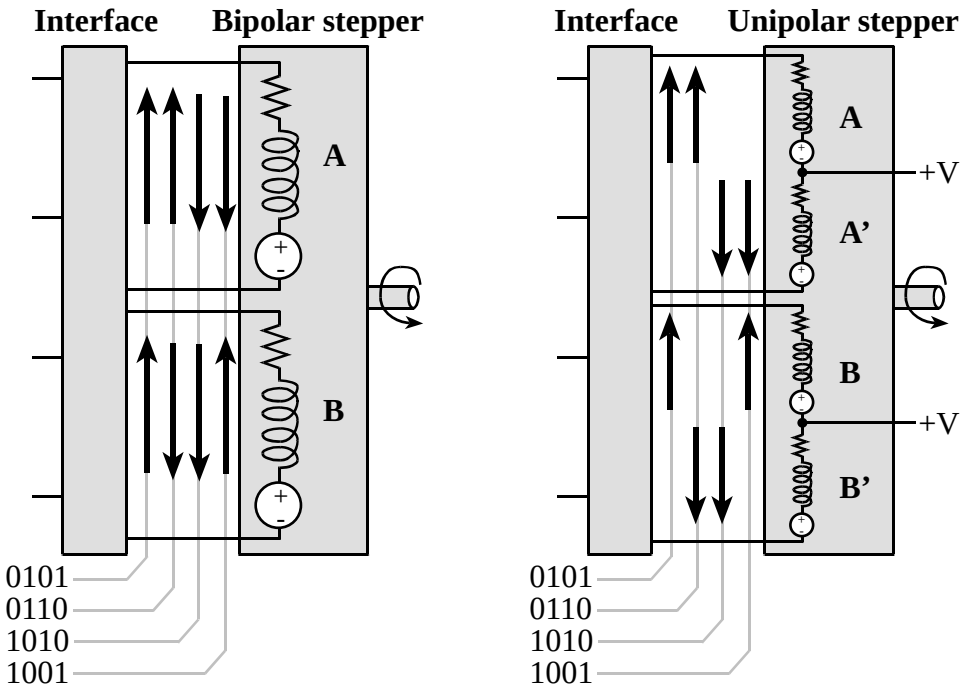
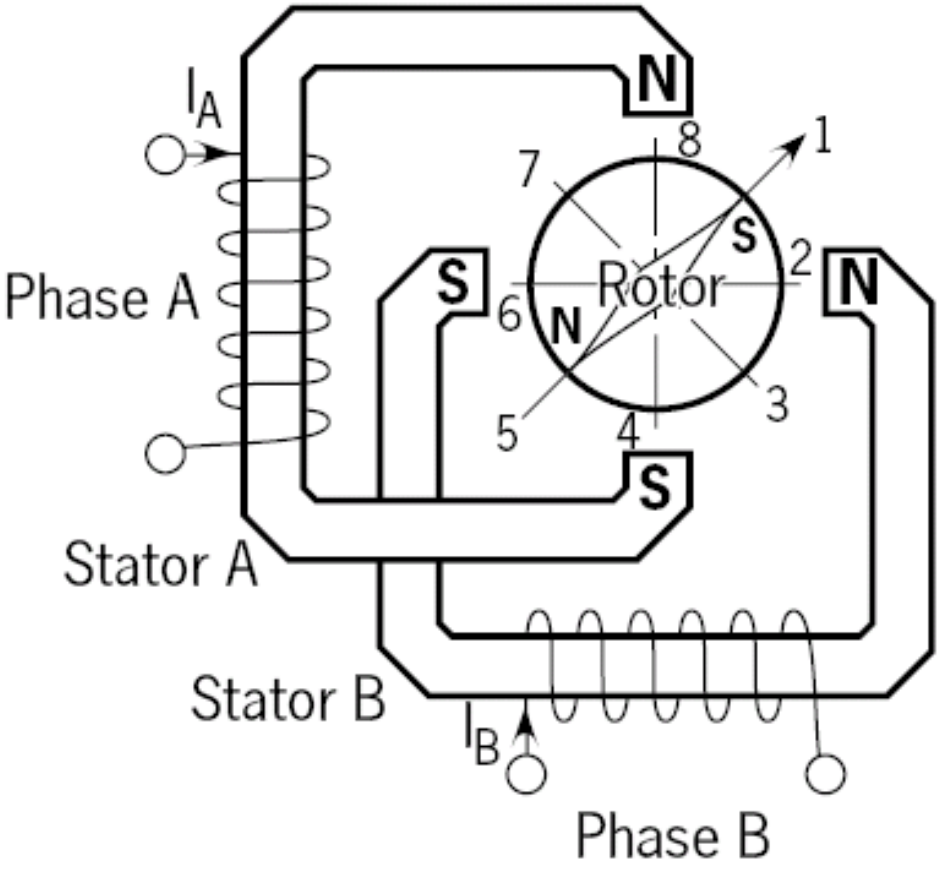
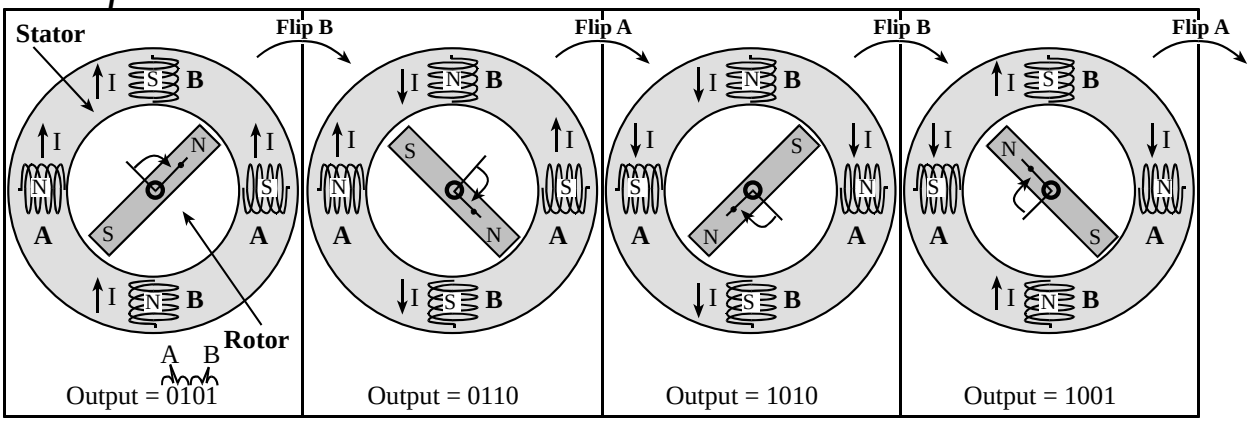


Figure 8.25. A bipolar stepper has 2 coils, but a unipolar stepper divides the two coils into four parts.



To rotate this stepper by 90^o, the interface flips the direction of one of the currents.



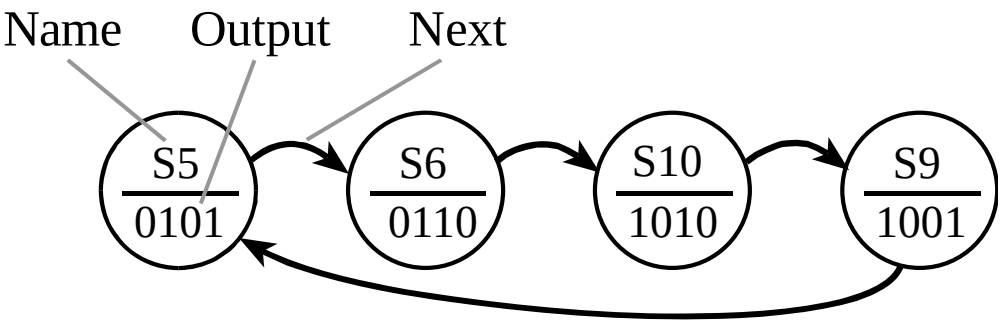


Figure 8.27. This stepper motor FSM has four states. The 4-bit outputs are given in binary.

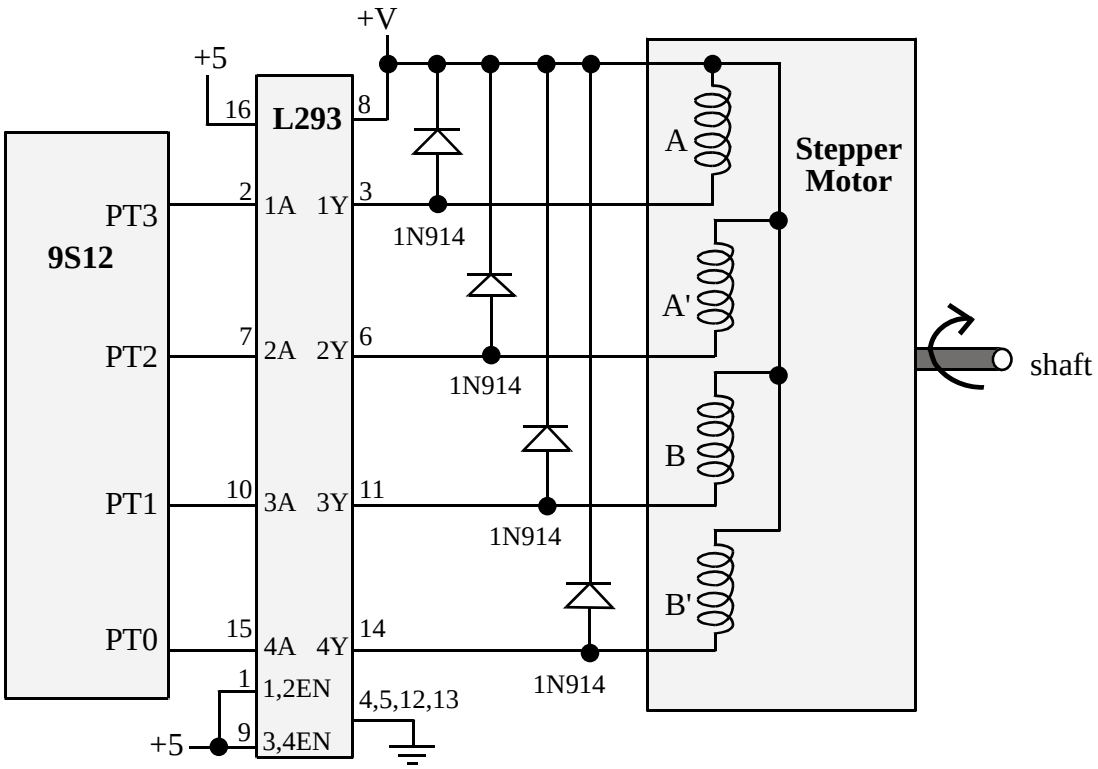


Figure 8.28. A unipolar stepper motor interfaced to a Freescale 9S12.

Speed = (1 rotation/**200** steps)*(1000ms/s)*(60sec/min)*(1step/**50ms**)
= 6 RPM

```
const struct State {
  unsigned char Out;          // command
  const struct State *next;}; // clockwise
```

```
typedef const struct State StateType;
#define S5 &fsm[0]
#define S6 &fsm[1]
#define S10 &fsm[2]
#define S9 &fsm[3]
StateType fsm[4]={
  { 5, S6},  // Out=0101, Next=S6
  { 6,S10},  // Out=0110, Next=S10
  {10, S9},  // Out=1010, Next=S9
  { 9, S5}}; // Out=1001, Next=S5
void main(void){ StateType *Pt;
  Timer_Init();
  DDRT = 0xFF; // outputs
  Pt = S5;     // initial state
  while(1){  // embedded systems never quit
    PTT = Pt->Out;     // stepper out
    Timer_Wait1ms(50); // 50ms wait
    Pt = Pt->next;     // Clockwise step
  }
}
```

Program 8.10. Stepper motor controller.

The **IO->Stepper…** command allows you to connect stepper motors.

Each stepper has four control wires.

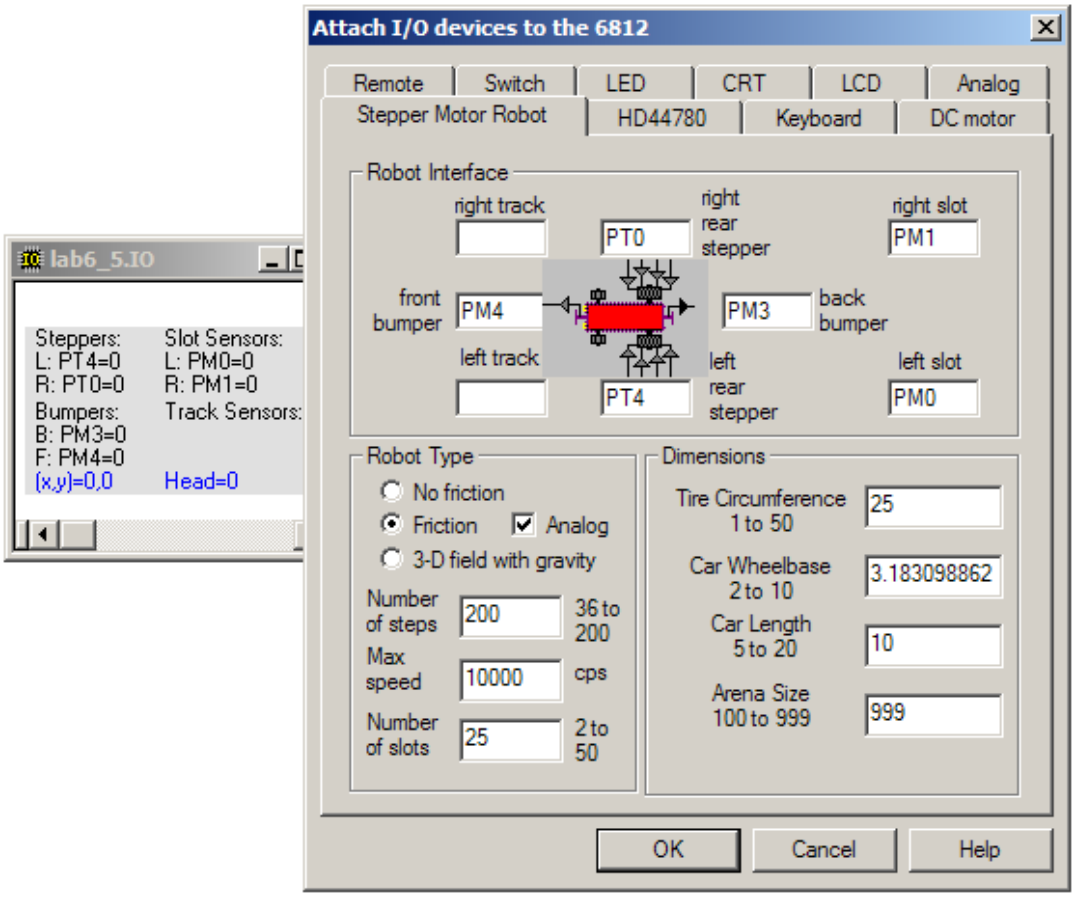
Each output within the sequence 1010, 1001, 0101, 0110, …
will cause the motor to make one clockwise step.

Each output within the reverse sequence 0110, 0101, 1001, 1010, …
will cause the motor to make one counterclockwise step.

There are a fixed number steps per rotation.

Lab 9

The two motors independently control the two drive wheels of a robot car.



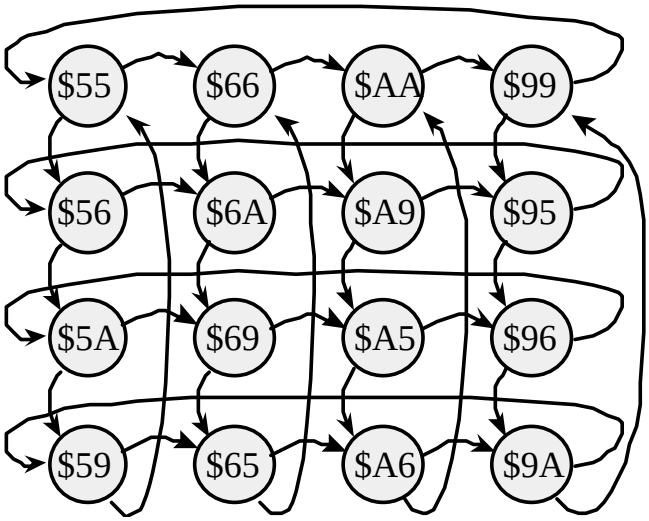
One possible way to configure the stepper motor car.

Command	Left track	Right track	Δθ (°)	Δx (m)	Robot motion
(F,F)	full-step forward	full-step forward	0	0.5	forward
(f,f)	full-step backward	full-step backward	0	-0.5	backward
(F,f)	full-step forward	full-step backward	-6	0	turn CW
(f,F)	full-step backward	full-step forward	6	0	CCW

Table 3. These four full-step commands are sufficient to move the robot.

Possible data structures

Only some arrows shown



Layered solution

- **Low level**
  - **Move forward**
  - **Move back**
  - **Turn 90 degrees CW**
  - **Turn 90 degrees CCW**
  - **Turn to North**
- **High level**
  - **Game algorithm**
    - **Search for opponents**
    - **Target and fire**
    - **Defense**
    - **Lifepacks**

Feedback

- **Read sensors**
  - **(x,y) position, ADC channels 0,1**
  - **Heading angle, ADC channel 2**