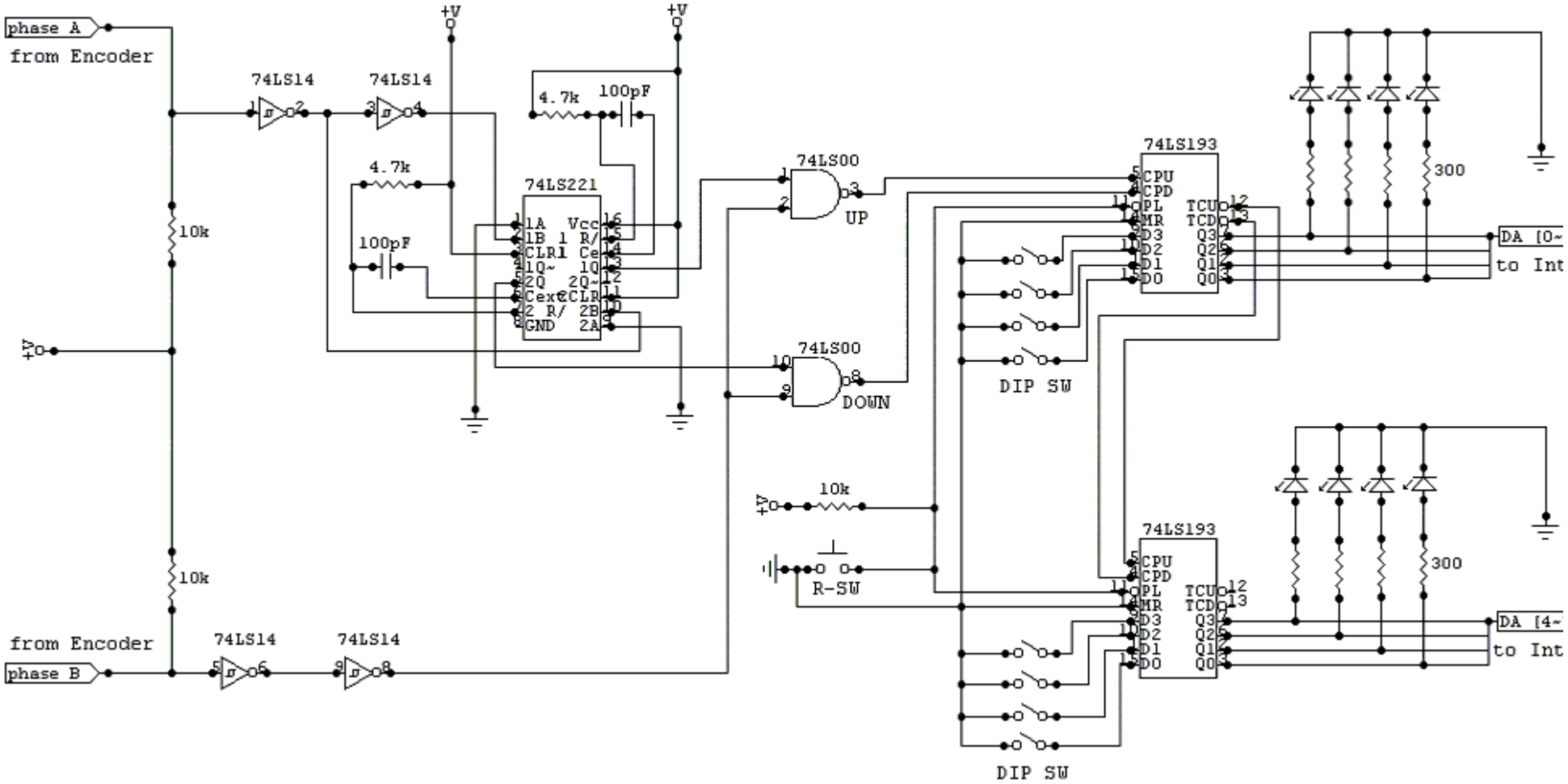
< 인크리멘탈 엔코더 읽기 방법 연구 >

엔코더에는 인크리멘탈식이나 엡솔류트식이 있고... 어쩌구 하는 이야기는 우선 생략합니다.

본 문서의 주요 목표는, Pocket-bot을 위해 인크리멘탈 엔코더를 가장 간단하면서 효과적으로 사용할 수 있는 방법에 대해 고찰해 보는 것입니다. 일단, 생각해 볼 수 있는 방법에 대해 소개해 봅니다.

1. 무식하게 로직으로 구현하기

이 방법은, 멀티바이브레이터(업/다운 에지에서 톡톡 튀는 출력 발생)와 NOT/NAND 게이트 따위를 엮어서 하드웨어적으로 구현해 보는 것입니다. 일단 회로도를 참고해 봅니다.



좀 너저분한 그림이라도 이해 바랍니다.

우선, 풀업저항 10k로 A 및 B 상의 입력이 들어오죠. 이 신호는, 슈미트 트리거를 거치면서 좀 깨끗하게(?) 만들어 줍니다. 먼저 A 상의 신호는, 원래 신호와 반전된 신호로 갈라져 각각 멀티바이브레이터인 74221로 입력됩니다. 여기에 달린 컨덴서와 저항은, 출력될 신호

의 듀티비를 정해 주기 위해서 조정된 값인데... 정확한 값까지는 필요없겠죠. 적당히 짧은 시간 동안 튀어주면 충분합니다. 암튼, 이로 인해 A 상의 신호는 업/다운 에지에서 각각 신호가 나오도록 변조(?) 비스무레하게 되겠죠. 한편, B상 신호는 그냥 들어오네요...

이제 NAND 게이트를 이용해서 연산 거치면, 보시다시피 UP/DOWN 신호로 딱 분리되어 나오겠죠. 이를테면, 엔코더가 오른편으로 돌때는 UP 신호선에서만 클럭이 챠르륵 나올테고, 왼편으로 돌때는 DOWN 신호선에서 클럭이 나올테구요.

74193은, 업/다운 4비트 바이너리 카운터죠? 그림의 회로도에서는, 8비트 카운터로 만들어 줄려고 캐스케이드 선을 두 개 달아붙여 연결했구요.

리셋 스위치는, 그림의 딥 스위치로 맞추어둔 설정값으로 강제로 리셋할 때 쓸려고 달아놓은 것입니다.

이런 카운터를 이용한다면, 8비트의 신호선을 MPU 입력단에다 연결해서 읽어 들이면 되겠군요. 그것도, MPU가 읽고 싶을때면 언제든지 읽기만 하면 되는 편리한 점이 있겠구요. 때문에 MPU의 리소스도 아껴질테구. 만일 PLD 등으로 위의 회로를 구현할 수 있다면, 하나의 칩으로 해결 될 수도 있겠군요(다만 74221 의 기능은 GAL로 굽는 것이 불가능 또는 힘들다고 알고 있습니다. 라이브러리에 없더군요).

근데 결정적인 단점이 발견됩니다.

8비트의 카운트 자릿수가 다 차 버리면, 오버플로우되어 문제가 발생되겠죠. 따라서, 이런 카운터는 회전수의 한계가 딱히 정해져 있는 관절 등의 경우에 유효한 것이라 할 수 있겠군요. 자율이동 로봇의 궤적 추적(?) 같은 것을 위해서는 훨씬 더 많은 자릿수가 필요할테니, 좀 보완이 필요할 것입니다.

만일 이런 회로를 이용해서 자율이동 로봇에 적용한다면... 다음과 같은 방법을 이용할 수 있겠다고 생각됩니다.

- 우선 74193 카운터의 리셋 스위치 부분을 MPU의 출력 비트 하나에 연결해 둡니다. 소프트웨어로 리셋해 주기 위해서...

- 카운터에서 나오는 바이너리 단자들은, MPU의 입력단에 연결해 두면 되겠군요.

- 이제 MPU는 내부 타이머를 이용해서, 일정시간마다 샘플링을 합니다. 이때, 읽은 값은 MPU의 레지스터 또는 RAM 상에 있는 변수에 저장하면 되겠구요.. 샘플링 직후에는 카운터에 리셋을 걸어 74193 카운터가 오버플로우 하지 못하도록 하면 되겠군요.

- 한편, MPU에 저장된 카운터 값은 적분 연산을 통해 현재의 로봇 위치를 구할 수 있겠고... 한편 샘플링 시간마다 얻어진 카운팅 값은 그대로 로봇의 현재 속도로 되겠군요.

현재 Pocket-bot 의 경우, 100RPM을 넘는 경우가 없을 것으로 사료되고 또한 엔코더 분해능이 33PPR 밖에 안되니... 샘플링 시간을 상당히 느리게 잡아도(심지어 1/20초에서도..) 카운팅된 바이너리 값이 4비트의 자리수조차 조차 넘지 않겠네요.

음...

엔코더를 2개 쓸거니깐, 하나당 4비트씩 할당하면 입력포트 8핀에... 리셋을 위한 출력포트 2핀만 있음 구현되겠군요. 덤으로 샘플링 타임(타이머 인터럽트)은 엄청 느려도 되고... MPU의 속도가 느린 편이고, 리소스가 부족한 상황에서 상당히 매력적인 솔루션이 될 수 있겠습니다...

(참고로, 샘플링 타임 수십 마이크로 세컨드 정도로 해서 PID 따위의 피드백 루프를 돌릴 경우, 386 정도의 MPU로는 모터 3개 정도의 제어가 한계라고 합니다. 실제로 해 본 것은 아니지만, 경험있는 분의 엔지니어링 센스이니까 어느정도 근거는 있는 예기라고 볼 수 있겠죠. 상용화된 고가의 모션 콘트롤 보드의 경우, 32비트 DSP와 MPU를 써서 6축 제어하는 정도는 보았습니다. 물론 이 경우 카운터 칩이 따로 나와있었죠.)

2. MPU로 직접 카운트하기

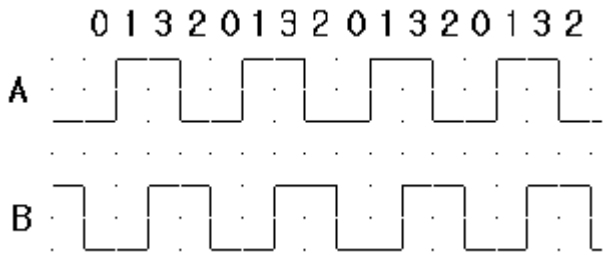
이런 방법은 MPU에 대해 어느정도 자신있는 분들이 가장 먼저 생각하는 것이겠죠. 예를들어, '엔코더를 어떻게 읽을까?'하고 질문하면, '당연히 MPU로 연산해서 읽어야지'라는 대답...

그 방법은 우선, 엔코더의 A 상과 B 상의 신호선을 MPU의 입력포트에다가 연결하는 것으로 시작해야 겠죠. 그담엔? 당연히, 엔코더가 뱉아낼 신호의 최고 주파수보다 어느정도 여유있을 정도로 더 빠른 샘플링 타임으로 주기적으로 읽어 들이는 것이 순서일 테고...

그렇게 하면, MPU는 엔코더 신호의 변화 과정을 실시간으로 모니터링 하는 것이 일단 가능해 지겠죠.

이제 이렇게 받아들여진 정보를 어떻게 조작하여 변수값을 +1 또는 -1 시켜 가느냐? 는 문제로 볼 수 있겠습니다.

우선 엔코더의 신호를 가만히 관찰해 보죠.



A와 B상의 90도 위상차로 인해 대략 위 그림과 같은 신호가 들어올 테고, 이를 10진수로 변환해 본게 그림 위의 숫자입니다.

그러니까, 엔코더가 정방향으로 돌때는 신호가 0132,0132,0132,... ... 하고 들어올 테고 역방향으로 돌때는 2310,2310,2310,... ... 하고 들어올테죠.

그럼, 정역 또는 정지 상태임을 판별하기 위해서는 간단히 테이블을 만들어 비교문 따위로 구분해 내면 되겠군요?

이를테면, 직전의 값이 0 인데 이번에 들어온 값이 1이면 정방향으로 1클럭 움직였다는 소리니깐... +1 시켜주면 되고... 이번에 들어온 값이 2이면 역방행으로 움직였다는 소리니깐 -1 해주면 되고. 간단하네요.

근데... 연산이 너무 길어지는 단점이 있군요(최소 8회 비교). 엔코더 따위 읽어들이는데 MPU의 클럭을 낭비할 수는 없다!는 생각이 듭니다.

그럼 이보다 간단한 방법은 없을까요?

0132 순서로 진행되는 코드는, 2비트의 그레이 코드라고 볼 수 있다고 합니다. 그럼 이놈을 0123 이라는 정상적인 순서의 바이너리 코드로 변환해 주면 될텐데, 그냥 그 변환 연산을 해 주면 되겠군요.

그 방법은 구체적으로... 샘플링 한 값의 bit0에 bit1로 XOR 연산해 주면 2와 3이 뒤바뀌는 결과가 되어 0123 순서의 바이너리코드가 된다고 합니다. 그럼 이 순서를 이용해서 정역 판정을 할 수 있겠고, 논리적으로 문제없이 해결되겠네요.

이 방법을 썼을 때, 예상되는 장점은... 우선 하드웨어가 엄청 간단해 지는군요. 잡스런 부품 다 없어지고, 달랑 와이어 두 개만 달아주면 되니...

다만, MPU에 연산 부담이 조금(?) 가중되고 샘플링 타임이 상당히 짧아져야 된다는 단점이 있습니다. 만일 MPU의 속도가 보장된다면 아무 문제 없으면서 가격이 다운되는 솔루션이 될 수 있겠군요.

< 결 론 >

외부 카운터를 달아붙여 구현하는 1번 방법은, 전체적으로 복잡성이 2번 방법보다 증가하고 있습니다. 따라서 2번 방법을 심각하게 고려할 만 하다고 보겠습니다.

4메가 헤르쯔의 8비트 MPU인 8515의 경우, 2번 방법을 사용하더라도 크게 무리는 없어 보입니다만... 현재 이 MPU가 감당해야 할 연산 요구가 이것에 국한 된 것이 아니므로 약간은 걱정도 됩니다. 하지만, 어차피 공부해 보자는 목적이니만큼 실제 구현해 보기 전에는 그

결과가 좋게 나올지 나쁠지에 상관없이 일단 부딪쳐 보는게 낫겠죠?

< 참고자료 >

1번 방법 : 제가 예전에 Inverted Pendulum 만들 때 써먹었던 방법임. 모 회사에서 산학으로 파견나와있던 공고출신 전자회로의 달인(?)이 슬쩍 알려주었었음.

2번 방법 : 일본인의 사이트인 http://www.elm-chan.org/ 에서 참조함. 물론 한미르로 번역해서 읽어 보았음.