



ARDUINO 특강

Beyond Design

강태욱

www.facebook.com/laputa999

laputa99999@gmail.com

2013-7-14

Revision history

Version	Description	Date	Author
0.1	지금까지 작업 내용 및 레퍼런스를 참고해 강의에 맞게 정리 함.	2013-07-14	강태욱

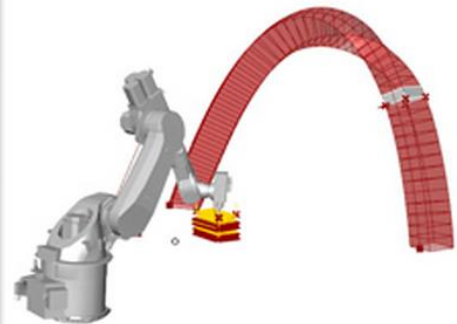
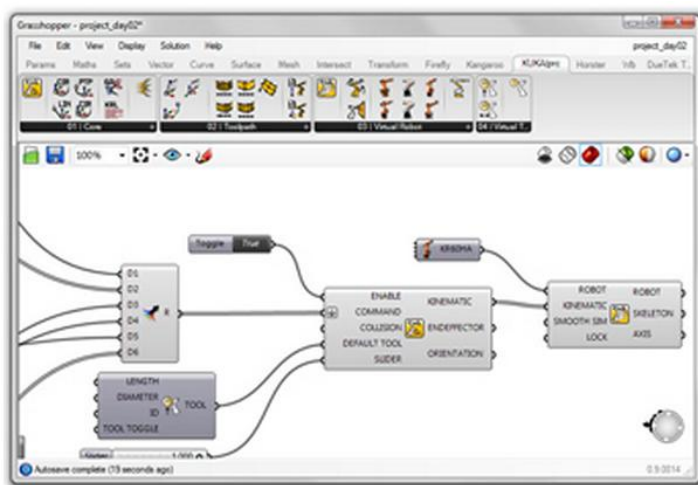
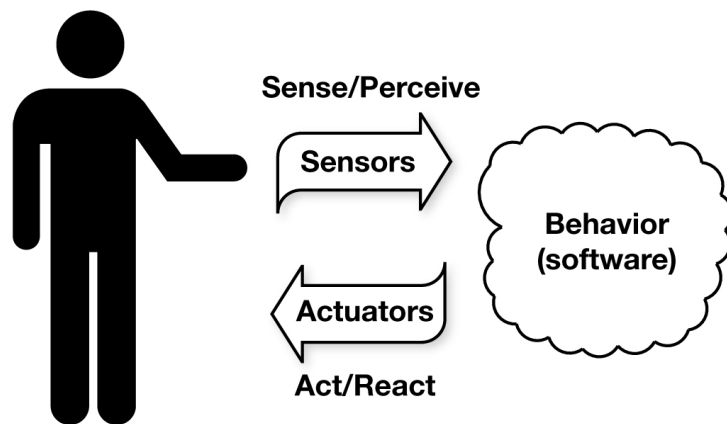
Table of Contents

1. 개요	5
2. 아두이노	6
2.1 개요	6
2.2 전자회로	10
2.2.1 개요	10
2.2.2 데이터시트	11
2.2.3 브레드보드	12
2.2.4 저항	13
2.2.5 LED	14
2.2.6 디바운스 회로	14
2.2.7 플립플롭과 래치	14
2.2.8 쉬프트레지스터	15
2.2.9 회로도 표시	16
2.2.10 릴레이	18
2.2.11 디스플레이	18
2.2.12 통신	18
2.2.13 기타 센서류	18
2.2.14 기타 모터류	19
2.3 전자회로 개발 준비	19
2.4 프로그래밍	19
2.4.1 개요	19
2.4.2 프로그래밍 구조	19
2.5 실습	21
2.5.1 LED 점멸	21
2.5.2 스텝모터 구동	22
2.6 결론	27
2.7 과제	28
3. 프로세싱	29
3.1 개요	29
3.2 내용	29
3.2.1 유연성	29

3.2.2	화면	29
3.3	실습	31
3.3.1	아두이노와 프로세싱 연결	31
3.3.2	시그널 그래픽 가시화 실습	32
3.4	과제	34
4.	FIREFLY	35
4.1	개요	35
4.2	내용	35
4.2.1	프로그램 설치	35
4.2.2	Firefly Firmata Sketch 업로드	36
4.3	실습	37
4.3.1	Basics	37
4.3.2	센서 데이터 입력	37
4.3.3	센서 데이터 그래픽 가시화	39
4.4	결론	41
4.5	과제	42
5.	결론	43

1. 개요

이 특강에서는 아두이노 보드에서 자료를 읽고 프로세싱이나 그래스호퍼를 이용해 스크린에 그래픽을 출력하는 데 초점을 맞출 것이다. 이 작업은 프로세싱 프로그램에 자료를 입력하고 아두이노 프로그래머가 그래픽적으로 센서 신호를 볼 수 있도록 할 수 있다. 이런 입력은 아두이노 보드에 붙일 수 있는 어떤 것이든 가능하며, 거리 센서부터 나침반이나 온도, 압력, 변위 등 센서, 로봇틱스 기기, 컴퓨터 비전 회로, 네트워크 메쉬 망까지 다양할 수 있다.



이 특강에 사용된 일부 그림은 아래 레퍼런스를 참고하였으며 아두이노에서 공식적으로 제공된 예제를 본 강의에 맞게 가공하였음을 밝힌다.

1. 아두이노 홈페이지
2. 프로세싱 홈페이지
3. 그래스호퍼 홈페이지
4. Firefly 홈페이지
5. 마시모 밴지, 2012.7, 손에 잡히는 아두이노, 인사이트
6. 채진욱, 2011.3, 아두이노 for 인터랙티브 뮤직, 인사이트
7. 케이시 리아스, 벤 프라이, 2011.3, 손에 잡히는 프로세싱, 인사이트
8. Interactive Prototyping – An introduction to physical computing using arduino, grasshopper, and firefly

2. 아두이노

2.1 개요

아두이노는 물리적인 세계와 디지털 세계를 효과적으로 연결할 수 있는 오픈 소형 하드웨어 플랫폼이다. 마이크로 컨트롤러 보드와 이를 실행할 수 있는 소프트웨어를 포함한 전자 프로토타이핑 플랫폼으로써 많은 분야에 기여하고 있다.

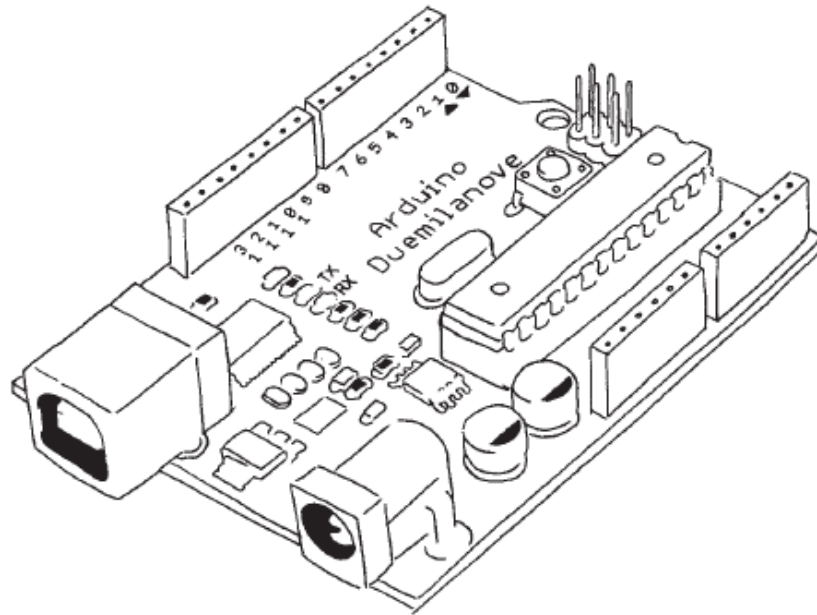
아두이노는 활용 목적에 따라 여러가지 보드 종류를 제공하고 있는 데 여기서는 아두이노 우노(ARDUINO UNO)를 사용할 것이다. 아두이노 우노는 아래와 같은 특징이 있다.

- 마이크로 컨트롤러: ATmega328
- 구동전압: 5V
- 입력 전압: 7~12V
- 디지털 입출력 핀: 14(이 중 6 개는 PWM출력)
- 아날로그 입력 핀: 6
- 입출력 핀의 DC전류: 40mA
- 3.3V 핀의 DC전류: 50mA

- 플래시 메모리: 32KB(이중 부트로더가 0.5KB 사용함)
- SRAM: 2KB
- EEPROM: 1KB
- 클록속도: 16MHz

이 단원에서는 당신이 아두이노 보드와 이를 사용할 수 있는 지식을 가지고 있다고 가정한다. 만약 그렇지 않다면, 먼저 <http://www.arduino.cc> 와 유명한 Getting Started with Arduino(Massimo Banzi 저, O'Reilly) 책을 이용해 학습을 해야 할 것이다. 기본적인 지식을 익혔다면, Making Things Talk(Tom Igoe저, O'Reilly)와 같은 책에서 프로세싱과 아두이노 사이에 데이터를 보내는 방법을 배울 수 있다.

데이터는 프로세싱 시리얼(Serial) 라이브러리를 이용해 프로세싱 스케치와 아두이노 보드 사이에 전달될 수 있다. Serial 은 한번에 한 바이트(byte)씩 데이터를 보내는 형식이다. 아두이노에서는 byte는 자료형이며, 0 에서 255 가지 값을 저장할 수 있다. 바이트는 int와 유사하지만 좀 더 작고, 큰 수를 전달해 주기 위해서는 바이트 열로 그 수를 쪼개 다음 나중에 재조합해야 한다.



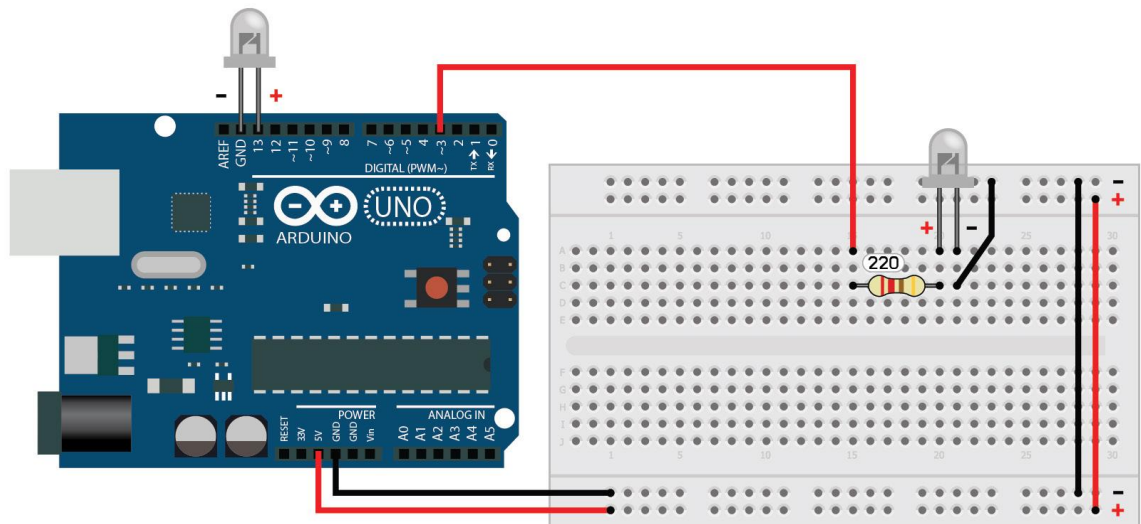
아두이노의 설치는 해당 홈페이지의 다운로드에서 받아 설치하면 되고 설치와 동시에 구동에 필요한 드라이버 등은 자동 설치된다.

컴퓨터와 아두이노의 통신은 COM포트를 통해 처리되는 데 보통 설치후 COM3 로 설정되어 있을 것이다. 이는 [도구][시리얼 포트]에서 확인할 수 있다. 참고로 [도구][보드]에서 현재 아두이노 보드를 설정할 수 있다.

아두이노는 아래와 같은 라이브러리를 지원하고 있다.



이제 아두이노를 실행하고 [파일][예제]에 있는 예제파일들을 확인해 본다. 다음은 LED 예제에 대한 회로도이다.



2.2 전자회로

2.2.1 개요

아두이노는 전자회로를 이용하므로 기본적인 전자회로는 알고 있어야 한다. 이 단원에서는 전자회로를 구성하는 데 필요한 몇가지 부품 및 구동 원리를 설명할 것이다.

각 소자는 소모하는 전류와 자체 저항 등을 가지고 있다. 이는 아래와 같은 옴의 법칙으로 계산할 수 있다.

$$V(\text{전압}) = I(\text{전류}) \times R(\text{저항})$$

1. 전압(Voltage)

전압과 전류의 차이는 수압과 물이 흐르는 것과 유사하다. 파이프에 있는 물을 어디론가 보내고 싶다면 수압을 주면 된다. 그 수압이 전압이라고 생각하면 된다.

만약 수압이 없다면 물은 파이프 안에서 그저 자유 운동을 하게 될 것이다. 우리가 흔히 보는 전선에는 전자가 마구 떠돌고 있는 데 그저 자유운동을 하고 있다고 보면 된다.

전압의 발생은 전위차에 의해 생기가 되는 데, 한쪽이 전압이 높고 한쪽이 전압이 낮다면 높은데서 작은데로 흐르게 된다. 그것이 바로 전압에 대한 개념이다.

전압에 단위는 Voltage 이다. 1v 는 1 줄(joule) 의 에너지가 1C(coulomb) 의 전자들을 움직일 수 있는 압력이다. 위의 C 는 단순한 전자들의 갯수를 의미한다.

2. 전류(Current)

전압이 전자에 줄 수 있는 압력이라면 전류는 전자의 흐름 흘러가는 것 그 자체를 이야기 한다. 전선에 전압을 주면 전자가 이동하는 데 그것이 바로 전류이다. 그냥 단순히 전자의 이동을 의미한다. 전류의 단위는 A,암페어(Ampere) 이다.

1A 는 1 초에 1C 만큼의 전자가 흐르는 것을 이야기 할 수 있다.

3. 저항(Resister)

전류의 흐름을 방해하는 요소이다. 저항이 너무 크다면 전류는 흐를 수 없지만, 저항이 하나도 없으면 전류는 매우 원활하게 흐르게 된다. 또한, 저항이 무지 클 때 전류를 흐르게 하고 싶다면 전압을 높이면 된다. 저항의 단위는 1 옴(Ohm) 이다.

1Ohm은 1v 를 전압에 걸렸을 때 1A 의 전류만 흐르게 하는 저항이다. 저항이 아무것도 없다면, 1A 의 전류만 흐르게끔 하는 저항이다.

저항이 전혀 없다면, 1v 의 전압을 줬을 때 무한대의 전류가 흐르게 된다. 그렇기에 저항이 전혀 없고 압력도 존재하니 미친듯이 흐르게 된다. 하지만 전선 또한 저항을 어느정도는 가지고 있다.

저항은 재질이 다른 전선을 왕창 꼬아놓은 것과 유사하다. 안에있는 전선 들이 좁고 길 수록 저항은 커지게 된다. 저항이 0 인 물체를 초전도체라고 한다(출처: 네이버 지식인).

2.2.2 데이터시트

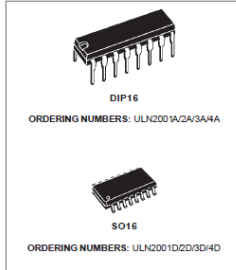
회로를 구성하는 반도체 칩을 포함한 소자의 동작 방식이나 기능을 기술해 놓은 매뉴얼을 데이터시트라고 하며 보통 아래와 같이 구성되어 있다. 어떤 소자를 사용하기 전에 사용법을 명확히 모르는 경우 데이터시트를 확인해 보면 된다. 기능, 동작 특성, 치수 및 크기, 회로 구성 등 다양한 정보가 포함되어 있다.



ULN2001A-ULN2002A ULN2003A-ULN2004A

SEVEN DARLINGTON ARRAYS

- SEVEN DARLINGTONS PER PACKAGE
- OUTPUT CURRENT 500mA PER DRIVER (600mA PEAK)
- OUTPUT VOLTAGE 50V
- INTEGRATED SUPPRESSION DIODES FOR INDUCTIVE LOADS
- OUTPUTS CAN BE PARALLELED FOR HIGHER CURRENT
- TTL/CMOS/PMOS/DTL COMPATIBLE INPUTS
- INPUTS PINNED OPPOSITE OUTPUTS TO SIMPLIFY LAYOUT



DESCRIPTION

The ULN2001A, ULN2002A, ULN2003 and ULN2004A are high voltage, high current darlington arrays each containing seven open collector darlington pairs with common emitters. Each channel rated at 500mA and can withstand peak currents of 600mA. Suppression diodes are included for inductive load driving and the inputs are pinned opposite the outputs to simplify board layout.

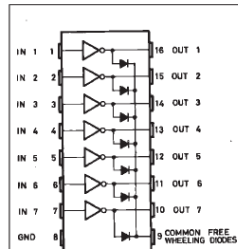
The four versions interface to all common logic families:

ULN2001A	General Purpose, DTL, TTL, PMOS, CMOS
ULN2002A	14-25V PMOS
ULN2003A	5V TTL, CMOS
ULN2004A	5-15V CMOS, PMOS

These versatile devices are useful for driving a wide range of loads including solenoids, relays, DC motors, LED displays, filament lamps, thermal print-heads and high power buffers.

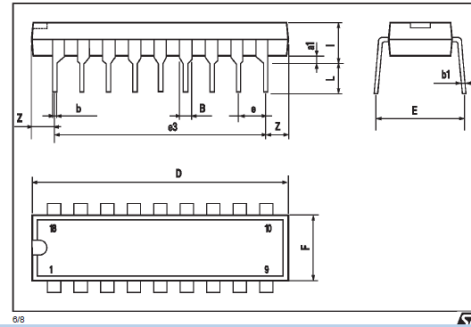
The ULN2001A/2002A/2003A and 2004A are supplied in 16 pin plastic DIP packages with a copper

PIN CONNECTION



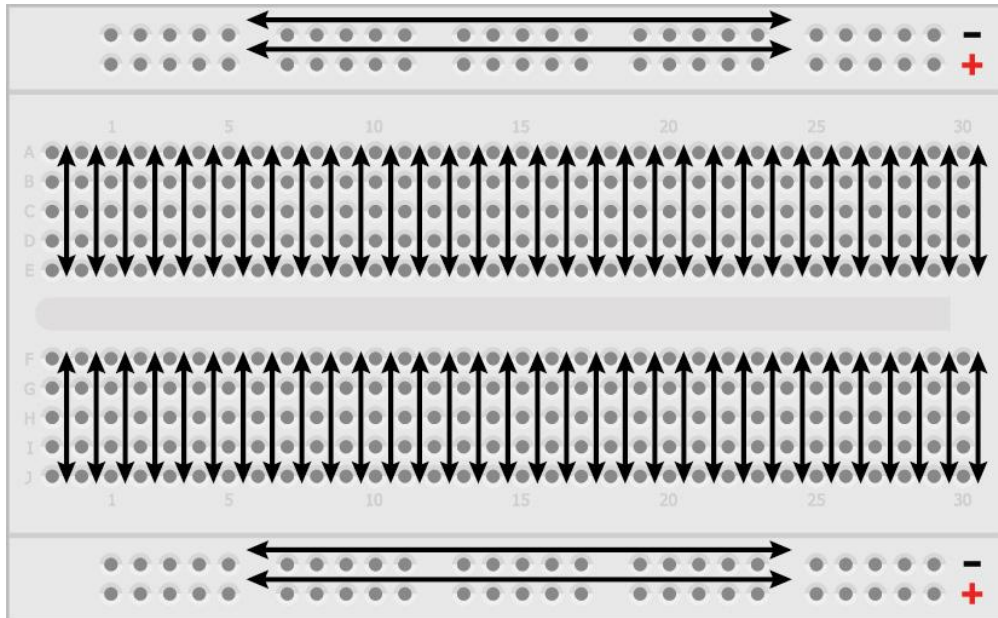
DIP16 PACKAGE MECHANICAL DATA

DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
a1	0.51			0.020		
B	0.77		1.85	0.030		0.065
b		0.5			0.020	
b1		0.25			0.010	
D			20			0.787
E		8.5			0.335	
e		2.54			0.100	
e3		17.78			0.700	
F			7.1			0.280
I			5.1			0.201
L		3.3			0.130	
Z			1.27			0.050



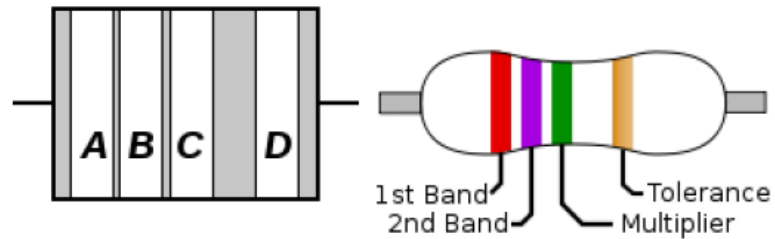
2.2.3 브레드보드

전자회로를 쉽게 만들고 구동하기 위해 필요한 부품으로 아래와 같은 모양의 회로를 가지고 있다.



2.2.4 저항

저항은 회로의 전압이나 전류를 제어할 때 활용한다. 저항 중 크기가 작은 것은 아래와 같이 색상코드로 저항값을 표시하고 있다.



아래는 색상코드이다.

Color	Digit 1	Digit 2	Multiplier	Tolerance
Black	0	0	$\times 10^0$	
Brown	1	1	$\times 10^1$	$\pm 1\%$ (F)
Red	2	2	$\times 10^2$	$\pm 2\%$ (G)
Orange	3	3	$\times 10^3$	
Yellow	4	4	$\times 10^4$	
Green	5	5	$\times 10^5$	$\pm 0.5\%$ (D)
Blue	6	6	$\times 10^6$	$\pm 0.25\%$ (C)
Violet	7	7	$\times 10^7$	$\pm 0.1\%$ (B)
Gray	8	8	$\times 10^8$	$\pm 0.05\%$ (A)
White	9	9	$\times 10^9$	
Gold			$\times 0.1$	$\pm 5\%$ (J)
Silver			$\times 0.01$	$\pm 10\%$ (K)

2.2.5 LED

빛을 내는 발광 다이오드이다. 한쪽 방향으로만 전류를 흘려보낸다. 전류가 흐르면서 발광부에서 빛이 나는 소자이다.

2.2.6 디바운스 회로

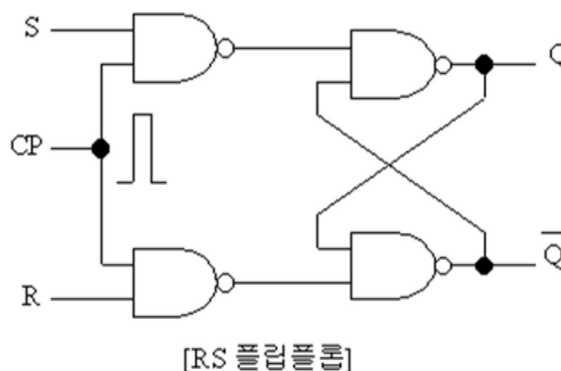
기계적인 스위치로 특정 소자를 구동할 때 진동으로 인해 전극판이 완전히 붙기 전까지 매우 짧은 시간동안 불안정한 신호 특성을 보인다. 이로 인해 마이크로 컨트롤러는 여러 번 누른것으로 인식할 수 있으며 이를 제거하는 것을 디바운스 회로라 하며 소프트웨어적으로도 제거할 수 있다. 이는 캐패시터를 포함한 74LS14 칩을 이용해 처리할 수 있다. 자세한 내용은 해당 칩의 데이터시트를 확인해 보도록 한다.

2.2.7 플립플롭과 래치

래치와 플립플롭은 두 개의 안정 상태를 갖는 일종의 기억 회로이다. 메모리 칩은 이와 같은 래치와 플립플롭이 고밀도로 집적된 것이다. 여기서 말하는 안정 상태란 회로의 외부로부터 입력을 가하지 않는 한 본래의 상태를 유지할 수 있는 상태를 말한다.

래치(latch)는 기본적인 플립플롭(basic flip-flop)을 말하며, 래치는 NOR게이트와 NAND게이트를 이용하여 구성할 수 있다.

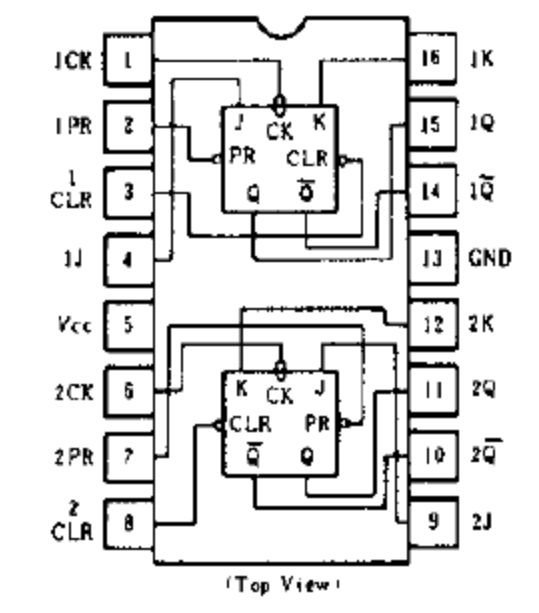
래치는 순차회로로서 기억소자로서의 기능을 수행 한다. 래치 종류 중에 SR 래치 회로는 출력 Q를 1로 세트시키거나 0으로 리셋 시키기 위해 S(set)와 R(reset)이라고 하는 입력단자를 갖는다.



R	S	Q
0	0	Q(이전값)
0	1	1
1	0	0
1	1	알수없음

플립플롭은 두 가지상태 사이를 번갈아 하는 전자회로이다. 플립플롭에 전류가 부가되면, 현재의 반대 상태로 변하며 (0 에서 1 로, 또는 1 에서 0 으로), 그 상태를 계속 유지하므로 한 비트의 정보를 저장할 수 있는 능력을 가지고 있다.

플립플롭의 종류에는 RS 플립플롭, D 플립플롭, JK 플립플롭, T 플립플롭 등 이있다.



레지스터에 기억된 내용을 이동하라는 지시 신호의 펄스가 하나씩 가해질 때마다 현재의 내용이 왼쪽이나 오른쪽의 이웃한 플립플롭으로 1 비트씩 이동되어 밀어내기와 같은 동작을 수행하는 레지스터를 직렬 이동(serial moving) 또는 시프트(shift) 레지스터라 한다.


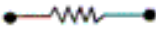


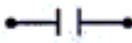








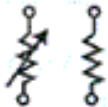






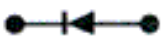





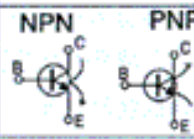

2.2.8 쉬프트레지스터

쉬프트레지스터는 클럭에 의해 동작되어 1 비트를 입력받아 차례대로 8 개의 플립플롭에 그 값을 설정하고 한꺼번에 출력할 수 있는 회로이다.

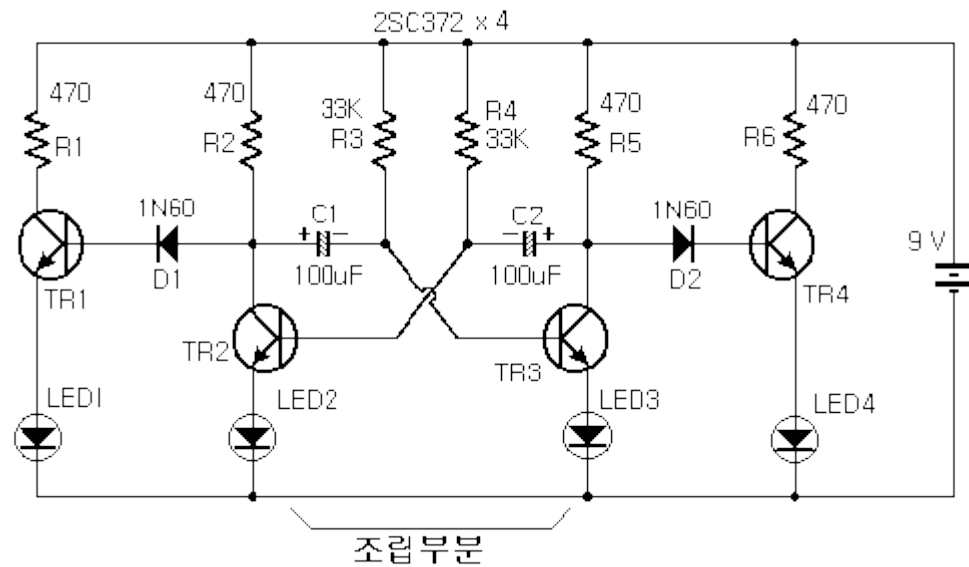
실제로 아두이노의 디지털 출력은 14 핀으로 이중 TX, RX를 제외하면 12 개이다. 16 개 출력을 하기 위해서는 다른 방법이 필요하다. 굳이 16 개 출력을 위해 아두이노를 하나 더 구매 하는 것은 비용 및 유지관리 면에서 바람직하지 않다. 이때는 래치나 플립플롭 회로가 담긴 회로를 사용하는 편이 좋다. 대표적인 것이 쉬프트 레지스터이며 대표적인 칩이 74HC595 이다. 이를 멀티플렉서 칩과 함께 응용하면 원하는 만큼의 LED를 구동할 수 있다. 이는 즉 원하는 만큼의 센서나 액추에이터를 제어할 수 있다는 의미와 같다.

2.2.9 회로도 표시

각 부품은 회로기호를 통해 다음과 같이 회로도에 표시된다.

부 품 명	기호	회로기호	부 품 명	기호	회로기호
저항		R		퓨즈	
콘덴서		C		소전구	
전해콘덴서		C		AC 플러그	
가변콘덴서		VC		전지	
볼륨		VR		전압계	
코일		L		전류계	
트랜스		T		다이오드	
스피커		SP		발광 다이오드	
스위치		SW		트랜지스터	
			집적회로	IC	

이를 이용해 다음과 같은 회로도를 표시한다. OrCAD와 같은 프로그램을 이용하면 회도로를 만들어 기판을 기판제작업소에 보내 만들 수 있으며 본인이 직접 제작할 수 있도록 프린트지도 인쇄할 수 있다. 더불어 시뮬레이터를 이용해 각 지점의 전압, 전류 특성 등을 미리 시뮬레이션하여 오작동을 미연에 막을 수도 있다.



2.2.10 릴레이

큰 부하를 다룰 때 사용하며 기계식과 전기식으로 스위치의 접점부를 온오프하는 소자이다. 보통 전기 전원을 끈다던지 전등 및 가전기기를 켜고 끄는 데 사용된다.

2.2.11 디스플레이

정보를 디스플레이하는 소자로 7-세그먼트, LCD(Liquid Crystal Display) 등이 있다.

2.2.12 통신

적외선 송수신 회로, 리모컨 회로, RFID리더, 지그비, GPS 등이 있다. 참고로 지그비와 같은 통신 모듈을 아두이노에 직접 장착할 수 있는 모듈 등이 개발되어 있다.

2.2.13 기타 센서류

기타 여러가지 센서 종류가 있으며 이는 디바이스마트 같은 전자부품 온라인 쇼핑몰 등에서 쉽게 구할 수 있다.

피에조 음향, 광 센서, 기울기 및 방향 센서, 온도 센서, 습도 센서, 방향 센서, 거리 측정 센서, 물체 감지 센서, 가속도 센서, 압력 센서, 변형율 센서 등이 있으며 각 센서별 출력 특성이 다르다. 그러므로 시그널을 입력할 때 출력 특성별로 정규화하는 과정이 필요하다. 또한, 센서에서 출력되는 전압은 불안전하게 튀는 신호가 포함되어 있으므로 안정화 처리할 필요가 있다.

2.2.14 기타 모터류

모터는 빠른 회전 속도를 가지고 있는 DC모터, 로봇 움직임 등을 제어할 때 사용하는 서보(Servo)모터, 방향/위치/속도를 완벽하게 제어할 수 있어 자동화에 많이 사용되는 스텝 모터 등이 있다.

2.3 전자회로 개발 준비

전자회로를 개발하기 위해서는 몇가지 준비물이 필요하다.

1. 테스터기: 회로나 소자의 전압, 전류 등을 측정할 수 있는 기기로 몇만원 이하로 구입할 수 있다.
2. 인두기 및 실납: 회로에 소자를 접합시키기 위해 납땜을 할 수 있는 도구이다.
3. 전선 및 니퍼(nipper)
4. 소자 부품 상자

기타로 오실로스코프가 있으면 좋다. 다만 가격이 비싸다.

2.4 프로그래밍

2.4.1 개요

아두이노는 자바를 기반으로 기기를 제어하는 라이브러리를 제공하고 있다. 그러므로 자바 프로그램의 기본적인 부분은 알고 있어야 활용이 가능하다.

참고로 예전에는 마이크로 컨트롤러를 제어하기 위해서는 기본적으로 C++과 어셈블리를 사용할 수 있어야 했다. 아두이노의 경우에도 Atmega328 칩으로 구동되는 데 예전에 이 칩을 사용하기 위해서는 저수준 언어를 익숙하게 다룰 수 있어야 했고 특별하게 개발된 개발도구로 플래쉬 메모리에 코드를 업로드해야 했다. 만약 에러가 나면 회로 및 코드 모두 지루한 디버깅과정을 거쳐야 했는 데 이는 저수준 언어와 입출력회로를 만들어야 했기에 에러가 발생되면 찾기가 어려웠기 때문이다.

2.4.2 프로그래밍 구조

아래는 아두이노에서 간단한 프로그래밍 구조이다.

프로그램에 기술된 스크립트 코드는 각 함수에서 위에서 아래로 실행된다.

setup() 함수는 아두이노에 의해 프로그램 시작시 한번만 자동으로 실행된다. 이 함수는 초기 프로그램 설정과 관련된 루틴을 실행하는 데 사용하려는 목적으로 제공된다.

loop() 함수는 아두이노에 의해 전원이 꺼지지 않은 한 무한히 자동 실행되는 함수이다. 이 안에 원하는 수행 로직을 코딩하면 된다.

```
#define LED 13    // 상수 정의

int val = 0;      // 정수형 전역변수 정의

void setup()
{
    pinMode(LED, OUTPUT);    // 13 번 핀을 출력핀으로 설정하도록 함수 호출
}

void loop()
{
    val = analogRead(0);      // A0 에서 신호를 읽음
    digitalWrite(LED, HIGH);  // D13 에 HIGH신호를 출력시킴
    delay(val);               // 읽은 신호만큼 동작을 지연시킴
    digitalWrite(LED, LOW);   // D13 에 LOW신호를 출력시킴
}
```

2.5 실습

2.5.1 LED 점멸

간단한 LED 점멸을 실습해 보기로 한다. 이를 위해 가변저항을 이용해 이에 따라 점멸 주기가 바뀌도록 프로그램해본다.

우선 아날로그 입력핀 A0 에 가변저항 2 번째 핀을 연결한다. 가변저항 1 번째는 5V, 가변저항 2 번째는 GND를 연결한다.

LED는 디지털 출력 D13 번과 GND를 각각 연결한다. D13 이 HIGH가 될때마다 LED는 켜질 것이다(*주: LED에 과부하를 방지하기 위해 저항을 사용해야 하나 가변저항을 사용해 부하를 줄였으므로 직접 연결한 것이다. 하지만 부하가 많은 부품을 직접 아두이노 보드와 연결하면 부품이 망가질 수 있으니 이때는 적절히 저항이나 다이오드를 연결해야 함에 주의한다).

```
#define LED 13
```

```
int val = 0;
```

```
void setup()
```

```
{
```

```
  pinMode(LED, OUTPUT);
```

```
}
```

```
void loop()
```

```
{
```

```
  val = analogRead(0);
```

```
digitalWrite(LED, HIGH);  
  
delay(val);  
  
digitalWrite(LED, LOW);  
  
delay(val);  
  
}
```

2.5.2 스텝모터 구동

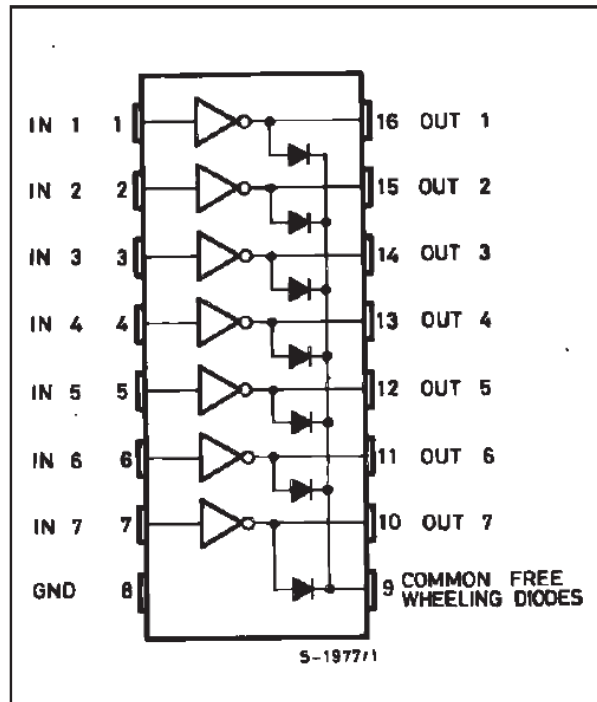
아래는 앞서 설명한 예제를 활용해 스텝모터를 구동하는 예제이다.

예제에서 사용할 스텝모터는 KHL35LL16K로 24V입력을 받을 수 있다. 데이터시트를 보면 6 개의 선이 있는 데 흰색과 회색은 Vcc 와 연결하고 나머지 오렌지색, 검정색, 노란색, 고동색은 시그널에 따라 모터를 회전시키는 데 사용된다.

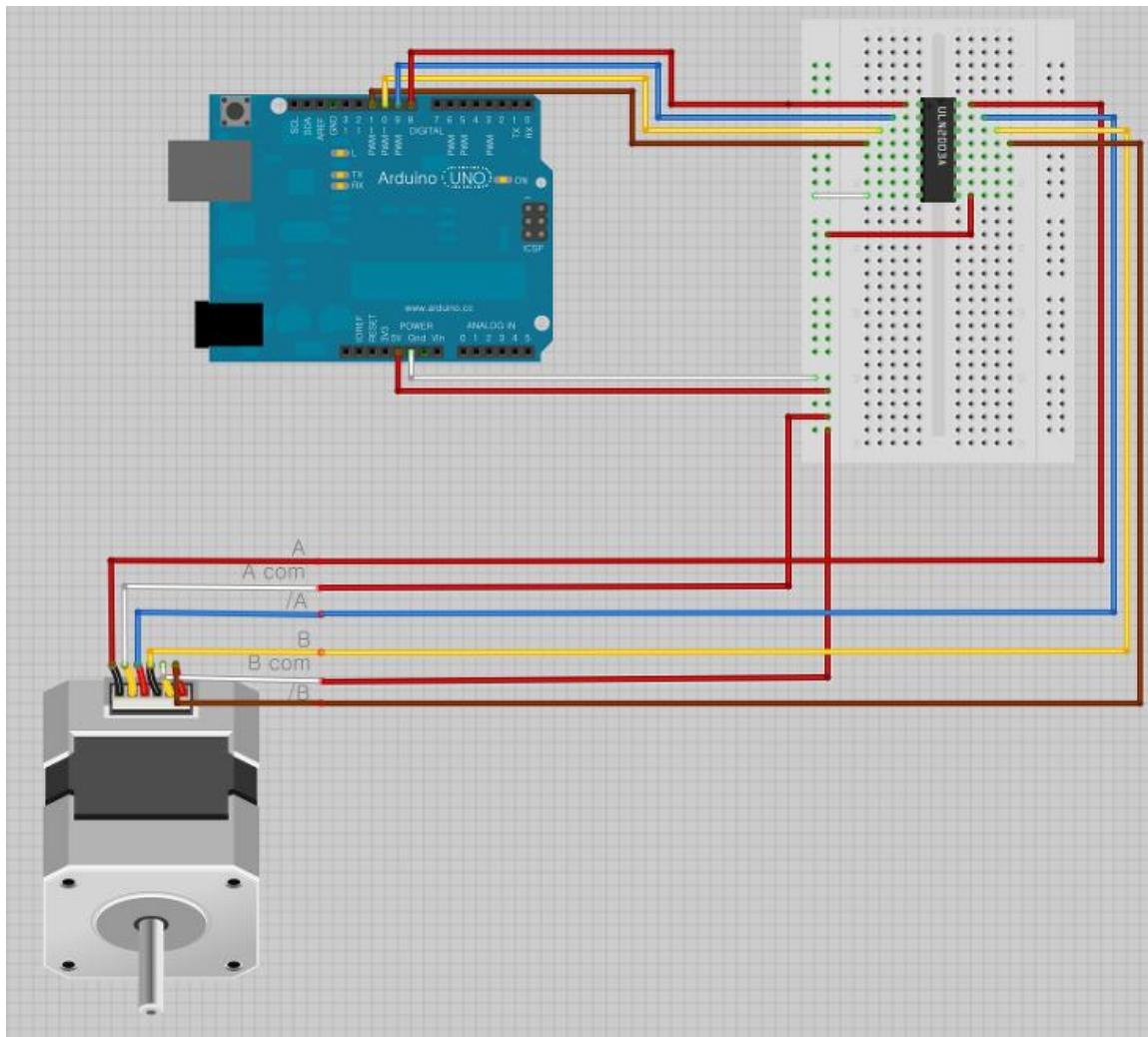
자세한 내용은 데이터시트를 참고한다.

스텝 모터를 구성하기 위해 ULN2003APG 칩을 사용할 것이다. 참고로 이 칩은 딜링톤회로로 구성되어 있으며 5V를 입력(in)하면 0V출력(out)되는 부품으로 마이컴을 24V의 전류로부터 보호한다. 딜링톤회로를 사용한 또 다른 이유는 많은 부하가 필요한 부품을 구동하기 위해서이다. 딜링톤 회로는 두개의 트랜지스터를 직렬로 연결시켜 큰 출력을 요구사하는 회로에 사용되며 이런 회로는 높은 구동 전압이나 전류를 요구하는 모터 등에 활용된다.

PIN CONNECTION



딜링턴회로와 스테핑모터를 이용한 회로도는 아래와 같다.



이 회로를 구동하기 위해 아래와 같은 코드를 입력해 본다.

```
/* Stepper Unipolar Advanced
```

```
* -----
```

```
* 
```

```
* Program to drive a stepper motor coming from a 5'25 disk drive
```

```
* according to the documentation I found, this stepper: "[...] motor
```

```
* made by Copal Electronics, with 1.8 degrees per step and 96 ohms
```


* per winding, with center taps brought out to separate leads [...]"

* [<http://www.cs.uiowa.edu/~jones/step/example.html>]

*

* It is a unipolar stepper motor with 5 wires:

*

* - red: power connector, I have it at 5V and works fine

* - orange and black: coil 1

* - brown and yellow: coil 2

*

* (cleft) 2005 DojoDave for K3

* <http://www.0j0.org> | <http://arduino.berlios.de>

*

* @author: David Cuartielles

* @date: 20 Oct. 2005

*/

```
int motorPins[] = {8, 9, 10, 11};
```

```
int count = 0;
```

```
int count2 = 0;
```

```
int delayTime = 500;
```

```
int val = 0;
```

```
void setup() {
```

```

for (count = 0; count < 4; count++) {
    pinMode(motorPins[count], OUTPUT);
}
}

```

```

void moveForward() {
    if ((count2 == 0) || (count2 == 1)) {
        count2 = 16;
    }
    count2>>=1;
    for (count = 3; count >= 0; count--) {
        digitalWrite(motorPins[count], count2>>count&0x01);
    }
    delay(delayTime);
}

```

```

void moveBackward() {
    if ((count2 == 0) || (count2 == 1)) {
        count2 = 16;
    }
    count2>>=1;
    for (count = 3; count >= 0; count--) {
        digitalWrite(motorPins[3 - count], count2>>count&0x01);
    }
}

```

```

    }

    delay(delayTime);
}

void loop() {
    val = analogRead(0);
    if (val > 540) {
        // move faster the higher the value from the potentiometer
        delayTime = 2048 - 1024 * val / 512 + 1;
        moveForward();
    } else if (val < 480) {
        // move faster the lower the value from the potentiometer
        delayTime = 1024 * val / 512 + 1;
        moveBackward();
    } else {
        delayTime = 1024;
    }
}
}

```

2.6 결론

아두이노는 최초로 5 만원 정도를 투자해 DIY가능한 대중화된 소형 컴퓨터 오픈 플랫폼으로써 다양한 라이브러리와 프로그램 연동을 통해 생각할 수 있는 모든 것은 개발할 수 있다.

최근에는 라스베리 파이, 비글본과 같은 더 강력해진 소형 컴퓨터가 판매되고 있으며 리눅스 OS등을 설치할 수 있어 응용프로그램 개발이 더욱 쉬워졌다. 또한, 홈시어터 정도는 쉽게 구동이 가능할 정도로 실용적이다.

해외에서는 Make 저널을 통해 일반인들이 만든 다양한 작품들이 소개되고 있으며 이 중에는 초등학교 2 학년 여학생이 만든 작품부터 아마추어 소형 인공위성까지 다양하다.

2.7 과제

1. 16 개의 LED를 제어해 보자.
2. 전압 = 전류 X 저항이다. 이 식을 이용해 여러가지 부하에 필요한 전압 등을 계산하기 위한 적절한 회로를 구해보자.

3. 프로세싱

3.1 개요

프로세싱은 이미지 애니메이션 그리고 그의 상호작용을 처리하기 위해 만들어진 소프트웨어이다. 프로세싱은 대화형 그래픽 작성을 통해 프로그래밍을 배울 수 있는 방법을 제공한다. 프로세싱에서 그래픽을 만들어내는 과정을 스케치라고 한다.

프로세싱은 특히 미디어아트나 알고리즘 기반 그래픽 형상 생성 연구 등에 많이 활용되고 있으며 심지어 액츄에이터나 컴퓨터 비전 등 그 활용범위가 매우 넓다.

3.2 내용

3.2.1 유연성

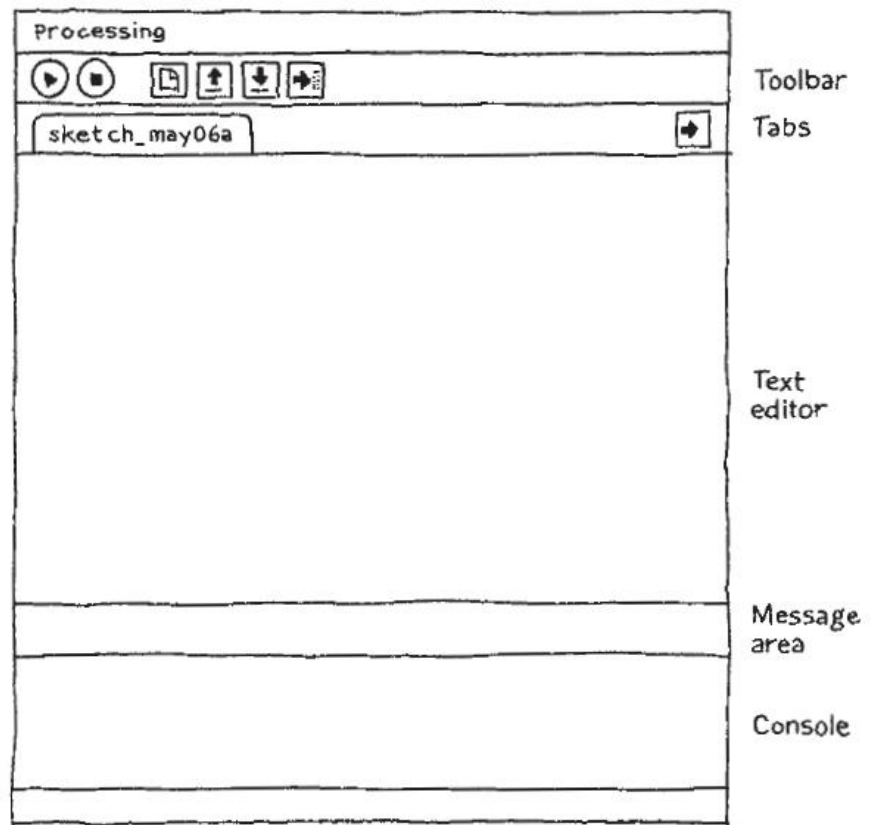
다른 소프트웨어 유틸리티와 마찬가지로, 프로세싱도 다른 조합 함께 다양한 도구로 구성되어 있다. 프로세싱은 한 줄일 수도 있고 또는 수천 줄 수 있기 때문에, 변화와 성장을 위한 여지가 충분이 있는 프로그램이다. 100 개 이상의 라이브러리는 프로세싱을 사운드, 컴퓨터 비전, 디지털 제조를 포함하는 도메인에 더욱 더 쉽게 응용프로그램을 개발할 수 있도록 한다.

3.2.2 화면

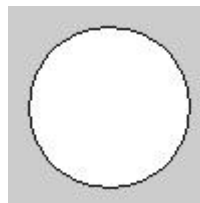
실행하면 다음과 같은 화면이 표시된다.



Display window



편집기에서 다음을 입력한다.



ellipse(50, 50, 80, 80);

이 코드의의미는"80 픽셀의 너비와 높이로, 중심이 왼쪽에서 오른쪽으로 50 픽셀 위에서 아래로 50 픽셀의 타원을 그린다는 뜻이다.

아래에 보이는 실행 버튼을 클릭한다.



제대로 다 입력했다면, 당신은 위의 타원 이미지를 볼 수 있을 것이다.

3.3 실습

3.3.1 아두이노와 프로세싱 연결

프로세싱과 아두이노를 시리얼 포트를 이용해 연결해본다. 이를 위해서는 먼저 아두이노에서 특정 아날로그 핀에서 신호를 읽어 시리얼포트로 전송해주는 간단한 코드를 설치해야 한다.

아래와 같이 코딩한 후 아두이노에 전송해 본다.

```
// Note: This is code for an Arduino board, not Processing

int sensorPin = 0; // Select input pin

int val = 0;

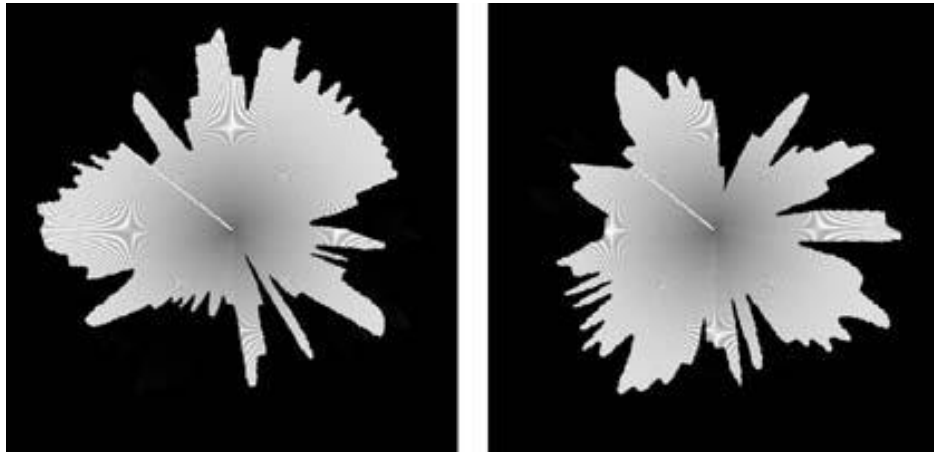
void setup() {
  Serial.begin(9600); // Open serial port
}

void loop() {
  val = analogRead(sensorPin) / 8; // Read value from sensor
  Serial.print(val); // Print variable to serial port
  delay(100); // Wait 100 milliseconds
```

```
}
```

3.3.2 시그널 그래픽 가시화 실습

다음과 같이 시그널을 입력받아 데이터를 그래픽으로 가시화하는 예제를 만들어 보자.



아래의 코드를 프로세싱에 입력해 본다.

```
import processing.serial.*;

Serial port; // Create object from Serial class
float val; // Data received from the serial port
float angle;
float radius;

void setup() {
  size(440, 440);
  frameRate(30);
  strokeWeight(2);
```



```

smooth();

String arduinoPort = Serial.list()[0];

port = new Serial(this, arduinoPort, 9600);

background(0);
}

void draw() {

  if ( port.available() > 0) { // If data is available,
    val = port.read(); // read it and store it in val
    // Convert the values to set the radius
    radius = map(val, 40, 60, 0, height * 0.45);
  }

  int middleX = width/2;
  int middleY = height/2;

  float x = middleX + cos(angle) * height/2;
  float y = middleY + sin(angle) * height/2;

  stroke(0);

  line(middleX, middleY, x, y);

  x = middleX + cos(angle) * radius;
  y = middleY + sin(angle) * radius;

  stroke(255);

  line(middleX, middleY, x, y);

  angle += 0.01;
}

```

}

3.4 과제

1. 오실로스코프는 가격이 최소 40 만원에서 1000 만원을 육박하는 전자 측정 장치이다. Processing을 이용해 간단한 소프트웨어형 오실로스코프를 만들어보자.

4. FIREFLY

4.1 개요

Firefly는 그래스호퍼와 아두이너 마이크로프로세스 간 브리지를 위해 개발된 소프트웨어 도구이다. 실시간으로 물리적 세계와 디지털세계간의 연결을 지원한다.

그래스호퍼는 강력한 파라메트릭 모델링을 지원하므로 이를 이용해 물리세계의 신호를 디지털 모델로 표현하거나 역으로 디지털 모델을 물리세계의 액추레이터와 같은 기기를 동작시킬 수 있다.

생성 모델링(Generative modeling)으로써 다양한 해석 시뮬레이션 가시화에서도 활용될 수 있으며 여러 하드웨어 장치들과 데이터를 주고 받을 수 있을 수 있다. 이러한 하드웨어 장치들 중에서는 빛, 온도 센서 뿐 아니라 모터 등 다양한 액추레이터를 지원한다.

4.2 내용

4.2.1 프로그램 설치

Firefly 툴바를 그래스호퍼에 설치하기 위해서는 아래아 같은 소프트웨어가 사전에 설치되어야 한다.

- Rhino 4.0 SR8 (evaluation version)
- Grasshopper v. 8.004 (or higher)

앞 두 소프트웨어가 설치되면 Firefly를 아래 링크에서 다운받아 설치한다.

- <http://www.fireflyexperiments/download> 에서 Firefly_Build 1.00xx.zip 파일을 다운받는다.

라이노와 그래스호퍼를 실행하기 전에 먼저 Grasshopper가 설치된 프로그램 폴더의 "Components Folder"에 방금 받은 압축파일을 풀어 복사한다. 그 파일 안에는 Firefly.gha, libTUIO.dll와 같은 실행에 필요한 파일이 포함되어 있다.

이제 라이노를 실행하고 명령창에서 "Grasshopper" 명령을 입력한다.

4.2.2 Firefly Firmata Sketch 업로드

그래스호퍼와 아두이노사이의 데이터 커뮤니케이션을 위해서 아두이노에 이를 위한 코드를 업로드해야 한다.

만약 아두이노 우노를 사용하고 있다면 아래 Firefly 에서 제공되는 파일 중에 "Firefly_UNO_Firmata"를 열어서 실행한다.

이제 아두이노에 설치된 이 코드를 통해 Firefly와 통신을 할 수 있다.

4.3 실습

4.3.1 Basics

만약 하나 이상의 아두이노와 컴퓨터를 연결한다면 가능한 포트들을 장치 관리자가 폴링하여 연결된 장치의 수를 리턴한다. 당신이 COM포트 번호를 확인하고자 한다면 Tools>Serial Port 메뉴를 아두이노 IDE에서 확인할 수 있다.

이제 다음으로 캔버스에 [Open/Close Port](Firefly>Arduino Boards>Open/Close Port)를 드래그&드롭한다.

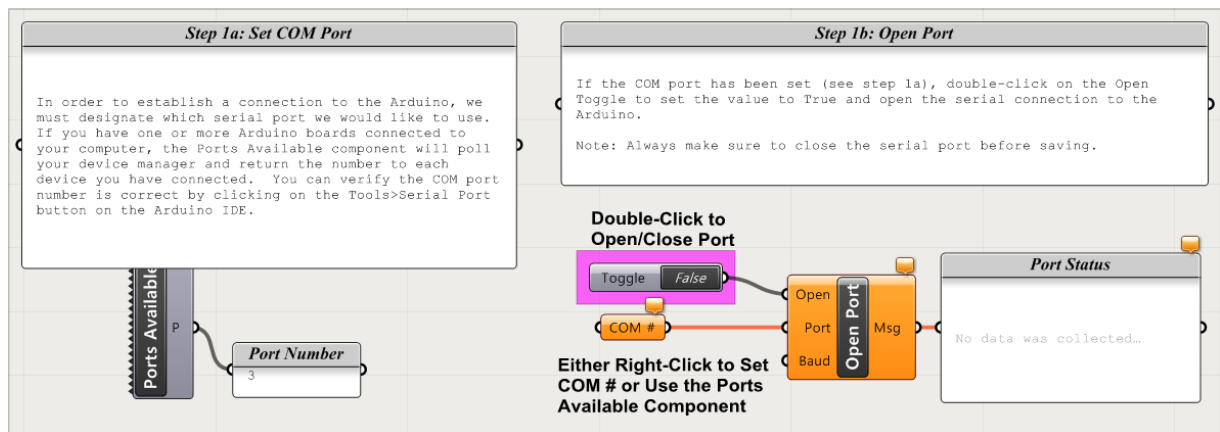
Open/Close Port 컴포넌트의 포트 입력에 포트 출력을 연결한다.

[Boolean Toggle](Params>Special>Boolean Toggle)을 캔버스에 드래그&드롭한다.

[Boolean Toggle]의 출력을 [Open/Close Port component]의 [Open input]과 연결한다.

[Boolean Toggle] 을 클릭해 True 값으로 설정한다.

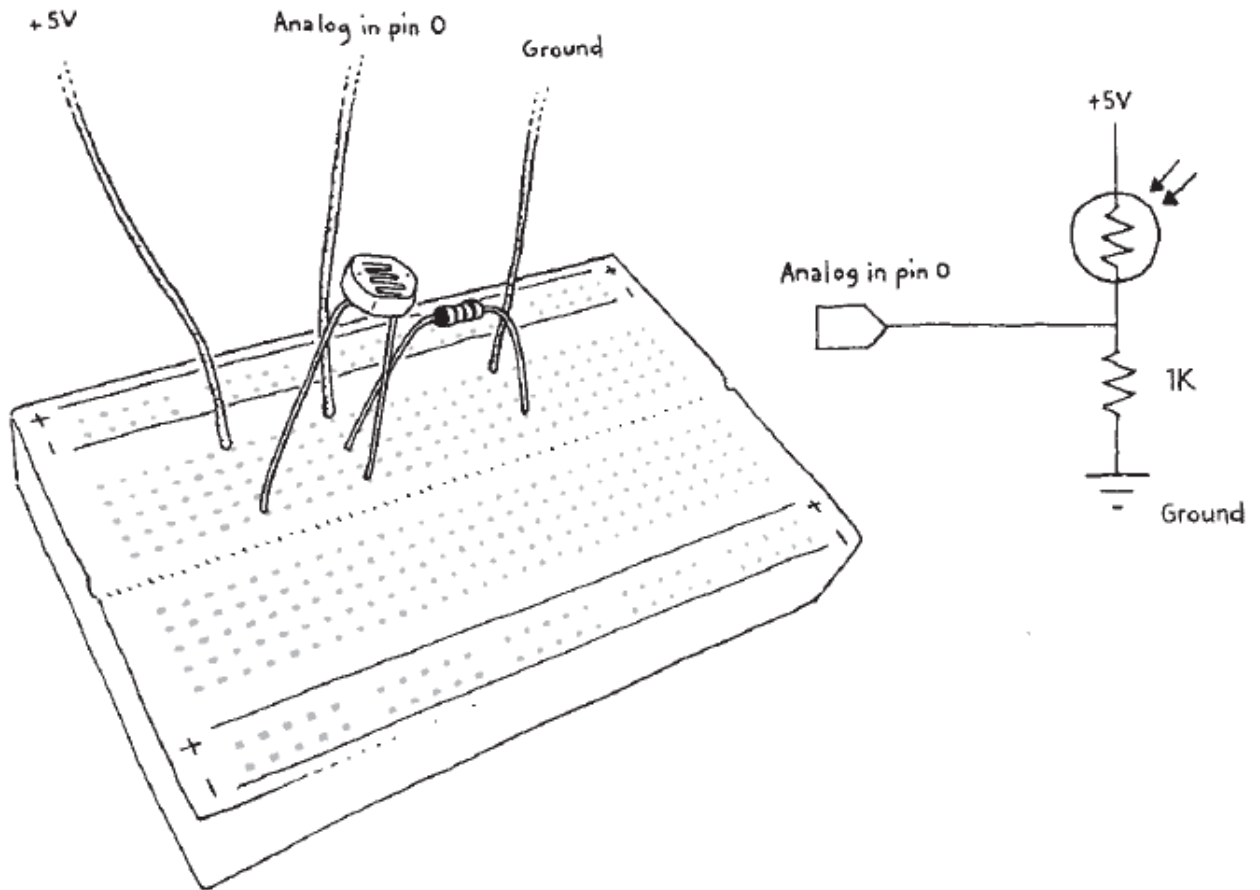
제대로 설정되었는 지 출력값을 확인해 보기위해 [Param][Input][Panel]을 두개를 생성해 각 컴포넌트의 [P]출력과 [Msg]출력에 연결해 본다. 제대로 설정되었다면 포트 번호와 시리얼포트가 오픈되었다면 메시지가 각각 출력될 것이다.



4.3.2 센서 데이터 입력

Firefly를 이용해 센서 데이터를 그래픽 가시화해 보도록 하겠다.

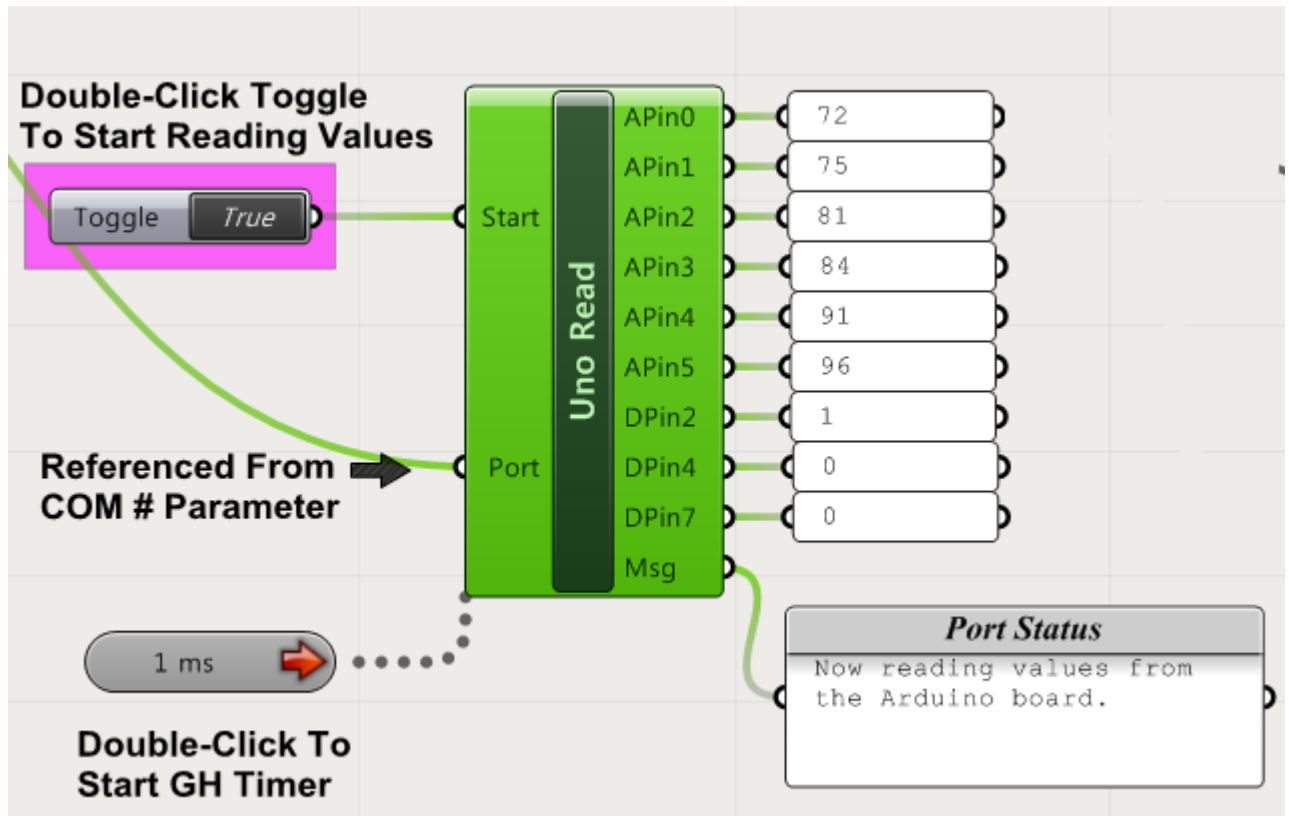
실습을 위해서 아두이노 A0 핀에 광센서가 아래와 같이 연결되어 있어야 한다.



1. [Arduino Boards][Uno Read]를 캔버스에 놓는다.
2. [Ports Available]컴포넌트의 [P]출력을 [Uno Read]의 [Port]와 연결한다.
3. [Boolean Toggle]을 캔버스에 놓는다.
4. [Boolean Toggle]을 [Uno Read]의 [Start]와 연결한다.
5. [Boolean Toggle]을 더블클릭해 [True]상태로 변경한다.
6. 각 컴포넌트의 출력값을 확인하기 위해 [Params][Input][Panel]을 캔버스에 생성해 각 컴포넌트의 출력과 연결한다. 각 아날로그 핀에서 값이 입력되고 있는 상태를 볼 수 있을 것이다.

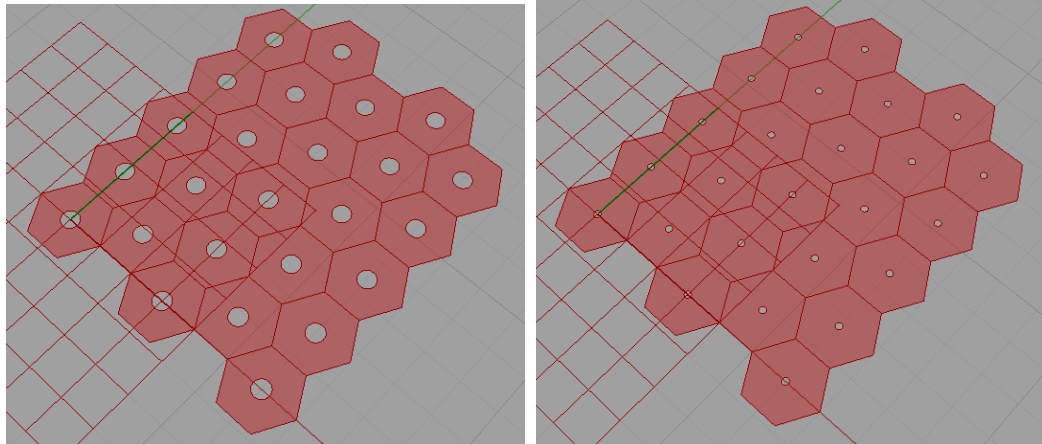
참고로 아두이노의 아날로그 출력은 0 에서 1023 사이의 값을 갖는다.

7. [Params][Util][Timer]컴포넌트를 캔버스에 놓는다.
8. [Timer]컴포넌트를 [Uno Read]와 연결한다.
9. [Timer]컴포넌트를 우클릭해 [Interval]메뉴에서 타이머 이벤트 발생 간격을 1 millisecond 로 설정하고 [Timer]컴포넌트를 더블클릭해 동작시킨다.

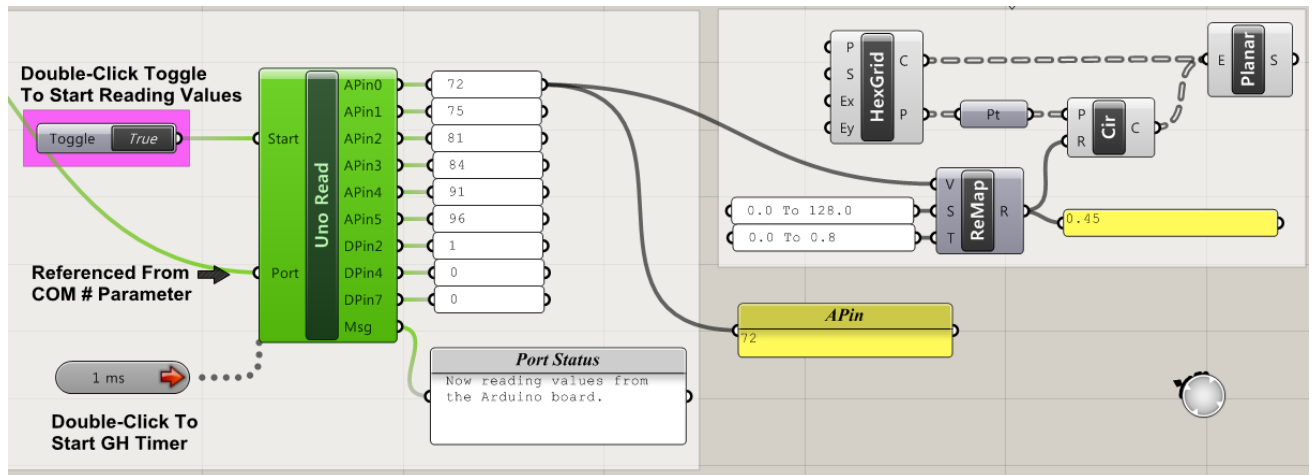


4.3.3 센서 데이터 그래픽 가시화

앞에서 얻은 센서 데이터를 그래픽 가시화해 보도록 하겠다. 이를 위해 6 각형 그리드 형상과 원을 이용해 아래와 같은 모양을 만들어 본다. 원은 센서 데이터에 따라 반경이 변하도록 한다.



1. [Vector][Grids][Hexagonal Grid]를 생성한다. 이 컴포넌트는 디폴트 값으로 25 개의 포인트가 이미 설정되어 있다.
 2. [Curve][Privimitive][Circle]을 생성한다.
 3. [Hexagonal Grid]의 [P]출력을 [Circle]의 [P]입력과 연결한다. 이 출력은 원의 중심점을 결정한다.
 4. 센서 데이터와 연동해 적절한 원의 반경을 설정하기 위해 약간의 스케일 계산이 필요하다. 원의 반경을 0 과 1.0 사이로 표현해 주기 위해 [Math][Operators][Division]을 생성한다.
 5. Apino0 와 [Division]의 [A]입력을 연결한다. 그리고 [B]입력에서 우클릭해 [Set Data Item]메뉴에서 1000 값을 입력한다. [Circle]의 [R]입력에 [[Division]의 [R]출력을 입력한다.
- 이제 원의 반경값이 센서에 의해 달라지고 있는 모습을 확인할 수 있다.
6. 헥사 그리드와 원을 조합해 서페이스를 생성하기 위하여 [Surface][Freeform][Planar Srf]를 생성한다. [Circle]의 [C]출력을 [Planar Srf]의 [E]입력으로 연결한다.
 7. 쉬프트키를 누른 상태에서 [Hexagonal Grid]의 [C]출력을 [Planar Srf]의 [E]입력으로 추가 연결한다.



이제 센서값에 의해 원이 변경되고 이에 따라 헥사그리드와 병합되어 생성된 서페이스가 라이노 화면에서 실시간으로 변하고 있는 그래픽을 확인할 수 있다.

4.4 결론

라이노의 그래스호퍼는 파라메트릭 모델링 기법을 직관적인 방법으로 지원하며 다양한 컴포넌트를 제공한다. 이러한 생성지향 모델링 기법은 다양한 곳에서 활용하고 있는 데 디자인, 비정형 건축 및 토목 모델링, 미디어 아트, 컴퓨터 기반 작곡 및 음향효과 제어, 로봇틱스, 프리페브리케이션, 엔터테인먼트 등 매우 다양한 영역에서 사용되고 있다.

이외에 다양한 지원 도구를 이용해 다양한 어플리케이션을 개발할 수 있으며 아래 프로그램을 설치하면 몇몇 추가적인 기능을 이용해 재미있는 작품들을 만들어 볼 수 있다.

1. WinAVR

WinAVR은 오픈소스 기반 개발 도구이다. Avr-gcc란 컴파일러와 avrdude란 도구를 포함한다.

- Go to: <http://sourceforge.net/projects/winavr/files/WinAVR/>

- Click on the folder called "20100110" and download the file WinAVR-20100110-install.exe
- Run the executable and follow the instructions
- Now, restart your computer

2. ReacTIVision

ReacTIVision 는 오픈소스 기반 컴퓨터 비전 프레임웍으로 fiducial 메이커와 finger 트래킹을 지원한다. TUI(tangible user interface)를 지원하기 위해 개발되었으며 다중터치 상호동작 서페이스 기반 어플리케이션 개발 등에 활용된다.

- Go to: <http://reactivision.sourceforge.net/#files>
- Download the reacTIVision vision engine for your appropriate platform (reactIVision-1.4-w32.zip (Win32))
- Unzip the file to a directory on your computer
- Run the reacTIVision.exe file to launch the vision engine
- To download the fiducial markers go to:
<http://reactivision.sourceforge.net/data/fiducials.pdf>

4.5 과제

1. Grasshopper의 다양한 형상들의 파라미터에 센서 데이터와 연동해 간단한 미디어아트를 만들어보자.
2. 간단한 스테핑 모터로 액추레이터를 제어해 보도록 하자.
3. 마이크로소프트 키넥트를 통해 Depth map을 통한 3D데이터를 취득할 수 있다. 연동하여 그래픽 가시화하는 프로그램을 만들어보자.

5. 결론

최근 들어 공학과 예술사이의 경계가 허물어지고 융합되어 사람들에게 즐거움을 주는 여러 다양한 결과물들이 만들어지고 있으며 꽤 비싼 값에 이러한 결과물들이 작품으로써 취급되는 현상이 나타나고 있다.

이는 미디어 아트뿐만 아니라 여러 다양한 분야에서도 나타나고 있는 것으로 하드웨어와 소프트웨어가 디지털 세계에서 융합되고 새로운 가치를 창출하는 일은 매우 흥미로운 일이며 특히 공학도가 스스로 DIY하면서 느끼는 엔지니어로서의 정체성을 확인하는 일은 큰 기쁨이 아닐 수 없다.

관련 기술이 표준화되고 대중화되고 있어 Open Lab과 같이 일반인들이 모여 전문가들 이상의 일을 하고 이런 작업들을 대중들이 열광하는 모습을 보면서 공유와 협업에 의해 사회적 의미를 만들어나가는 개인 패브릭케이터가 많아질 것이라 생각한다.

