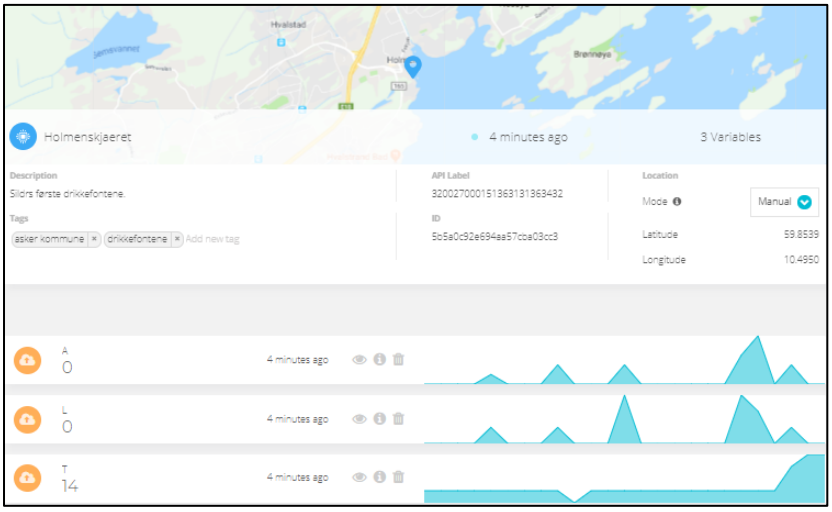
FONTENE

ELEKTRONIKK

Komponenter (Essensielle)	Kommentar	Illustrasjon
Knapp med LED	(12VDC)	
Particle Electron	(Shield PSU)	
Shield shield	(12VDC inn / 3,3&5VDC ut)	
Isolert LM35-temperaturmåler	(3V3DC)	
Spole til Magnetventil	(12VDC)	
HRI-Pulsgiver til Volummåler	(5VDC)	
Trafo 230VAC til 12VDC 2000mA	(x2)	
12V-3V3 og 5V-til-12V-krets	(Spennings-shield)	

DATA

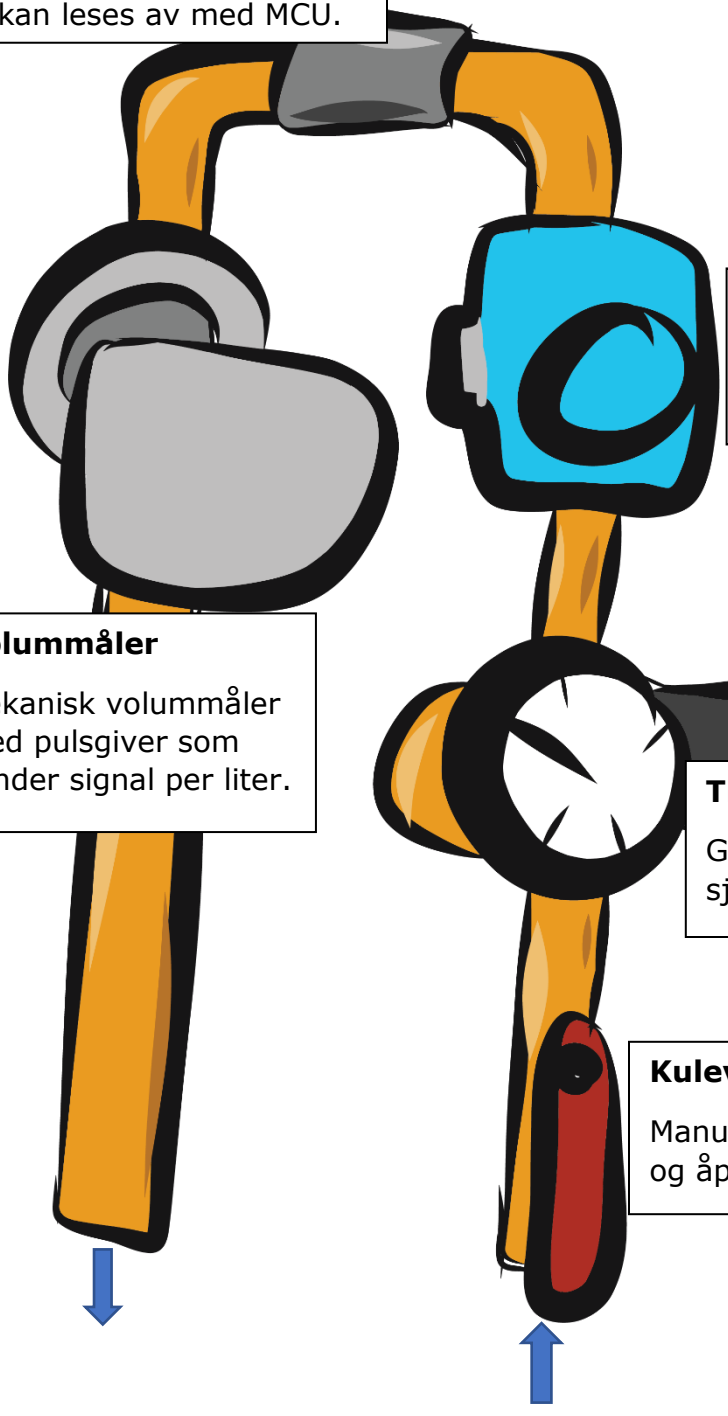
Particles økosystem + Ubidots



RØRSYSTEMET

Temperaturmåler

Motstandstemperatursensor som kan leses av med MCU.



Magnetventil

For å åpne og lukke vannkrets med MCU.

Volummåler

Mekanisk volummåler med pulsgiver som sender signal per liter.

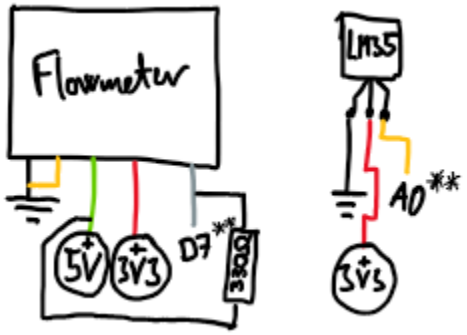
Trykkreduseringsventil

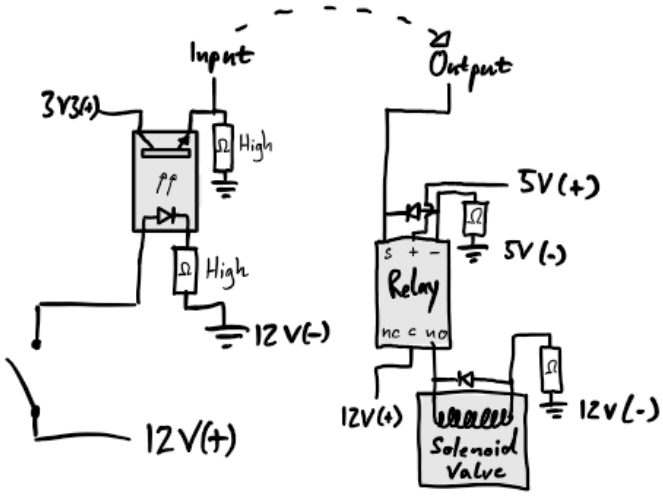
Gir muligheten til å sjekke og justere trykk.

Kuleventil

Manuell måte å stenge og åpne vannkrets.

KRETS


Flowmeter
5V
3V3
D7**
330Ω
LM35
A0**
3V3
** on shield,
not electron

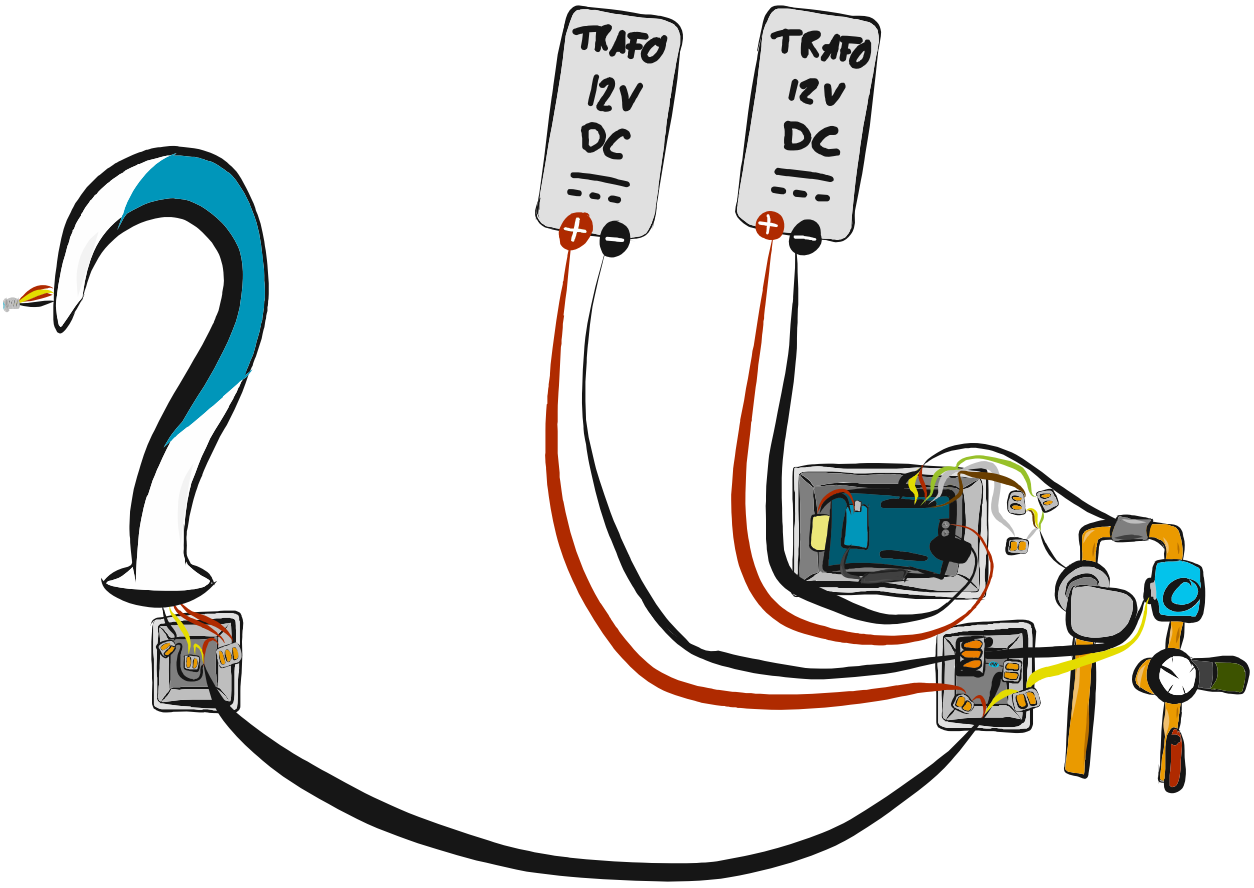

Input
Output
3V3(+)
High
ff
High
12V(-)
12V(+)
5V(+)
5V(-)
S + -
Relay
nc c no
12V(+)
Solenoid Valve
12V(-)

ELDRE UTSTYRSLISTE

Detaljert utstyrsliste for tidligere montering av elektronikk (med linker):
Obs! Ikke oppdatert og mangler mye fra det endelige systemet.

Komponent	Bestillingslink
1x Knapp med LED (12VDC)	N/A
1x Particle Electron + Batteri og Antenne	https://goo.gl/oApP6P
1x Shield shield, ferdig koblet (12VDC)	https://goo.gl/drFK1q
1x Koblingsboks, til krets	https://goo.gl/dZbSaz
2x Koblingsboks, til kobling	https://goo.gl/fWvyiv eller lignende
1x LM35 (temp., isolert) (3V3DC)	https://goo.gl/oxxfFz
1x Isoleringsteip, 10cm til LM35	https://goo.gl/bfVyDZ’
1x Haug med strips	Hvor som helst
1x Rull med elektrikerteip	Hvor som helst
1x Kjølepasta til LM35	https://goo.gl/WLp9uc
1x HRI-Pulsgiver til Volummåler (5VDC)	https://goo.gl/HKDmbT
1x Spole til Magnetventil (12VDC)	https://goo.gl/Bu9Fb7
1x Prototypeledninger	https://goo.gl/HRGYvu
1x Rød ledning 0,75, 10m kveil	https://goo.gl/iE8GaE
1x Sort ledning 0,75, 10m kveil	https://goo.gl/UVFJau
1x Gul ledning 0,75, 10m kveil	https://goo.gl/iE8GaE
2x Trafo 230VAC til 12VDC 2000mA	https://goo.gl/XFcFeu
8x Wago, Dobbel	https://goo.gl/QMU94G eller lignende
2x Wage, Trippel	https://goo.gl/efpxx4 eller lignende
(Alternative koblinger fra Biltema:	https://goo.gl/ZdqCLe)
1x Ledning med 3 ledere*	https://goo.gl/MU5rit eller lignende
1x 1K-Motstand e.l.	Hvor som helst

ELDRE TEGNING AV SYSTEM



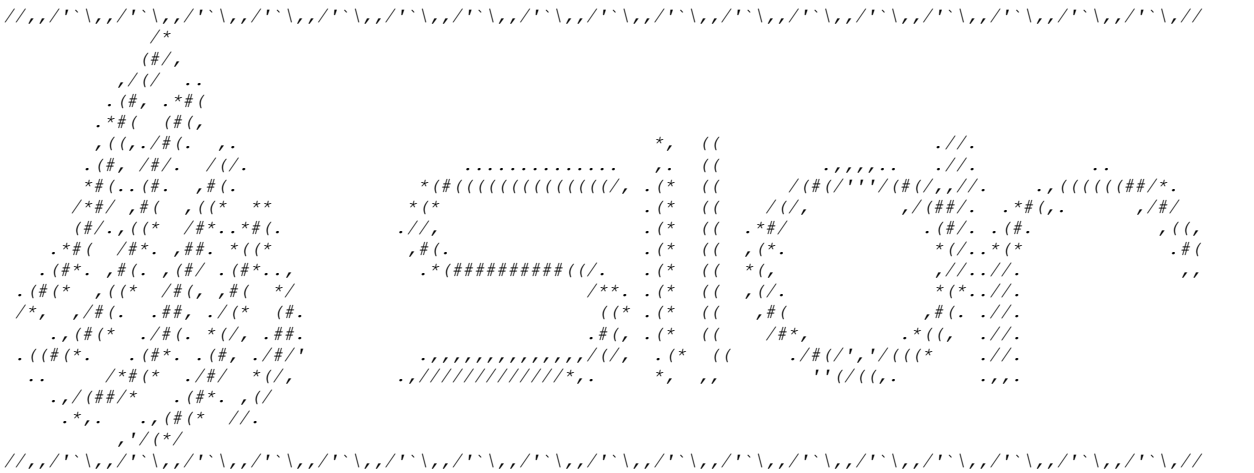
PROBLEMER

- **Fukt**
  - Knapp
    - Har foreløpig blitt løst ved å gradvis øke debounce
  - Temperatursensor
    - Har opplevd mistenkelig høye temperaturer en ca. måned etter installasjon
    - Skal byttes til vannsikker pt100 (om jeg husker rett)
- **Monterings- og sammensetningstid**
  - System og montering bør strømlinjeformes
    - Kretsen skal designes til ett kretskort med intuitiv I/O
    - Krets og rørsystem skal integreres til ett
    - Skal være ferdig koblet før montering
- **Overgang mellom spenninger**
  - Bruker foreløpig hjemmesnekret overgang mellom spenninger
    - Ønsker å utbedre løsning dersom det finnes åpenbart bedre løsninger.
- **Strømforbruk**
  - Ingen oversikt over effekt
    - Må forbedres hvis overgang til batteri eller vannkraft som strømtilførsel
- **Knotete kode**
  - Skyldes at vi er ferske
    - Skal skrives på nytt med leselighet og redigerbarhet i fokus

PLANER/IDÉER

- Gjøre system robust
- Gjøre system enkelt å reparere for en gjennomsnittlig elektriker/rørlegger
- Intuitiv fjernstyring av fontener (justere enkle parametere og skru av og på)
- Automatiske feilmeldinger
- Automatiske software-fixes (f.eks. ved åpenbar støy i knapp og temperatur)
- Vann som strømtilførsel
- Touchknapp (med hardware fra f.eks. Microchip, men krever mye utvikling), se siste del av dokument.
- Automatisert analyse og fremstilling av data
- API for kunder som ønsker å behandle tall selv

KODE



```
//LIBRARIES
#include <Ubidots.h>
Ubidots ubidots("BBFF-HNmFokn0E81qqRQwXxBjI05L2pF1tD");

//SYSTEM
SYSTEM_MODE(SEMI_AUTOMATIC);
SYSTEM_THREAD(ENABLED);

//PINS
const byte button_pin       = A1;
const byte valve_pin        = D6;
const byte flow_pin         = D7;
const byte temperature_pin  = A0;

//VARIABLES
//Valve
bool new_activate = true;
bool new_activate_count = true;
bool valve_first_if_statement_done = false;
int activate_count = 0;
unsigned long activate_time_interval = 7000;
unsigned long activate_time_initial = millis();
unsigned long activate_time;
unsigned long before_current = millis();
unsigned long before_initial = millis();
unsigned long valve_debounce_before = 200;

//Flow
bool new_flow = true;
bool new_pulse = true;
unsigned long new_flow_current;
unsigned long new_flow_initial;
unsigned long before_initial_flow;
unsigned long before_current_flow;
int flow_count = 0;

//Temperature
unsigned long temperature_time_initial = millis();
unsigned long temperature_interval = 5000;
int temperature_counts = 0;
int celsius_average;
float temperature_reading;
float celsius_sum = 0;
int temperature_calibration = -6;

//Publish
bool new_publish = true;
unsigned long publish_interval = 10*60*1000;
unsigned long publish_window = 10000;
```

```
//,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,//

/*_____
 / ____|     _
| (___   ___| |_ _   _ _ __
 \___ \ / _ \ __| | | | '_ \
 ____) |  __/ |_| |_| | |_) |
|_____/ \___|\__|\__,_| .__/
                       | |
                       |*/

void setup() {

    pinMode(button_pin, INPUT);
    pinMode(valve_pin, OUTPUT);
    pinMode(flow_pin, INPUT);
    pinMode(temperature_pin, INPUT);

    if (!Particle.connected()) {
        Particle.connect();
    }

    delay(1000);
}

/*_
 | |
 | |     ___   ___  _ __
 | |    / _ \ / _ \| '_ \
 | |___| (_) | (_) | |_) |
 |______\___/ \___/| .__/
                   | |
                   |*/

void loop() {
    valve();
    publish();
    flow();
    temperatureAnalog();
}

//,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,//

/*_       __  _                        =()=
 \ \     / /  | |              ,/'\_||_
  \ \   / /_ _| |_   _____     ( (___   `.
   \ \/ / _` | \ \ / / _ \     `\./  `==’
    \  / (_| | |\ V /  __/           |||
     \/ \__,_|_| \_/ \___|         \*|||*/

void valve() {

    int button_press = digitalRead(button_pin);

    activate_time = millis() - activate_time_initial;
    if (button_press) {
        if (new_activate) {
            before_initial = millis();
            new_activate = false;
        }
        before_current = millis() - before_initial;
        if (before_current >= valve_debounce_before) {
            if (new_activate_count) {
                activate_count += 1;
                new_activate_count = false;
            }
            digitalWrite(valve_pin, HIGH);
            activate_time_initial = millis();
            before_initial = millis();
        }
        valve_first_if_statement_done = true;
    } else if (activate_time < activate_time_interval && !button_press && valve_first_if_statement_done) {
        digitalWrite(valve_pin, HIGH);
        new_activate = true;
        before_initial = millis();
    } else {
        digitalWrite(valve_pin, LOW);
        new_activate = true;
        before_initial = millis();
        valve_first_if_statement_done = false;
        new_activate_count = true;
```

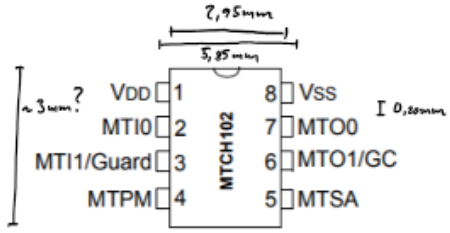
```
    }
}

void activatePublish() {
    print("A", activate_count);
    activate_count = 0;
}

/*_______ _
 |  _____| |                    // //
 | |__    | | _____   __       ||| |||
 |  __|   | |/ _ \ \ /\ / /     \\\ \\\
 | |      | | (_) \ V  V /      ||| |||
 |_|      |_|\___/ \_/\_/       //  */

void flow() {

    int flow = digitalRead(flow_pin);

    if (!flow) {
        if (new_pulse) {
            before_initial_flow = millis();
            new_pulse = false;
        }

        before_current_flow = millis() - before_initial_flow;

        if (before_current_flow >= 70 && new_flow) {
            flow_count += 1;
            new_flow = false;
            new_flow_initial = millis();
        }
    } else {
        new_flow_current = millis() - new_flow_initial;

        if (new_flow_current >= 54) {
            new_flow = true;
            new_pulse = true;
        }
    }
}

void flowPublish() {
    print("L", flow_count);
    flow_count = 0;
}

/*_______                                                          ,*'*,
 |__   __|                                  _                     | --|
    | | ___ _ __ ___  _ __   ___ _ __ __ _| |_ _   _ _ __ ___    | $-|
    | |/ _ \ '_ ` _ \| '_ \ / _ \ '__/ _` | __| | | | '__/ _ \   | $-|
    | |  __/ | | | | | |_) |  __/ | | (_| | |_| |_| | | |  __/ ,* ,$, *,
    |_|\___|_| |_| |_| .__/ \___|_|  \__,_|\__|\__,_|_|  \___| |  *@*  |
                     | |                                       *,___,*
                     |*/

void temperatureAnalog() {
    if (millis() - temperature_time_initial >= temperature_interval) {
        temperature_reading = analogRead(temperature_pin);

        float mv = (((temperature_reading + 1) / 4095.0) * 3300.0);
        float celsius = (mv / 10) + temperature_calibration;

        temperature_time_initial = millis();
        temperature_counts += 1;

        celsius_sum += celsius;
        celsius_average = celsius_sum / temperature_counts;
    }
}

void temperaturePublish() {
    print("T", celsius_average);
    temperature_counts = 0;
    celsius_sum = 0;
}
```

```
/*_____       _     _ _     _
 |  __ \     | |   | (_)   | |        ,*,
 | |__) |   _| |__ | |_ ___| |__    ,*   *,
 |  ___/ | | | '_ \| | / __| '_ \    |   |
 | |   | |_| | |_) | | \__ \ | | |   |   |
 |_|    \__,_|_.__/|_|_|___/_| |_|   |__*/

void publish() {
    if (new_publish && millis() % publish_interval <= publish_window && millis() > publish_window) {
        if (!Particle.connected()) {
            Particle.connect();
        }
        activatePublish();
        flowPublish();
        temperaturePublish();
        ubidots.sendAll();
        new_publish = false;
    } else if (millis() % publish_interval > publish_window) new_publish = true;
}

//,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,,/'`\,//

/*_____       _                 ______
 |  __ \     (_)       _       |      \
 | |__) | __ _ _ __ | |      | </>   |
 |  ___/ '__| | '_ \| __|   | ----- |
 | |   | |  | | | | | |__    | ----- |
 |_|   |_|  |_|_| |_|\__/   |______*/

void print(char* name, int value) {
    ubidots.add(name, value);
}
```

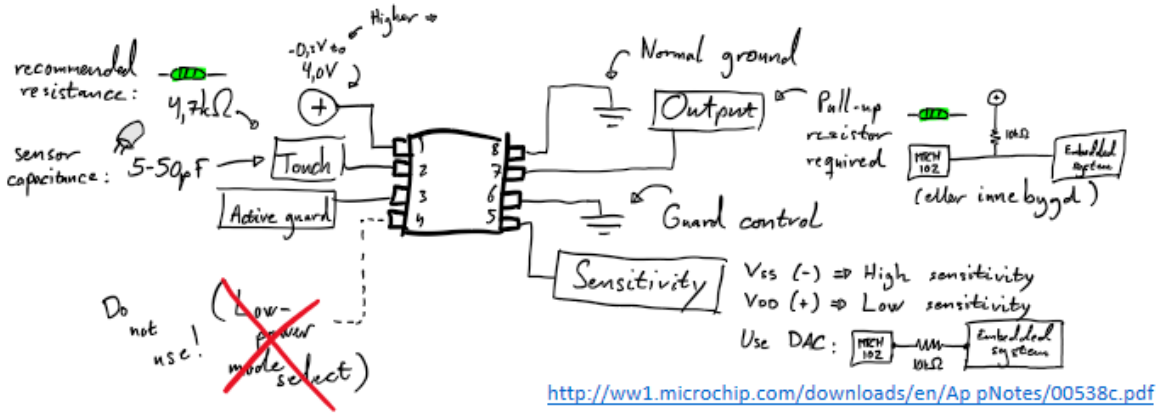
Extra: TOUCHKNAPP

MTCH102:

http://ww1.microchip.com/downloads/en/DeviceDoc/40001793C.pdf
http://no.farnell.com/microchip/mtch102-i-ms/proximity-touch-controller-2-ch/dp/2500436?ost=mtch102&ddkey=http%3Ano-NO%2FElement14_Norway%2Fsearch



VDD	1	Power Supply Input
MTI0	2	Proximity/Touch Sensor 0 Input
MTI1/Guard	3	Proximity/Touch Sensor 1 Input/Active Guard
MTPM	4	Low-Power Mode Select
MTSA	5	Sensitivity Adjust Input
MTO1/GC	6	MTI1 Detect Output (Active-Low)/Guard Control
MTO0	7	MTI0 Detect Output (Active-Low)
VSS	8	Ground



http://ww1.microchip.com/downloads/en/Ap pNotes/00538c.pdf

Touch button & Active guard:

- Plassere komponenter ✓
- Ikke plassere komponenter ✗

(or increase sensor size)

- <3mm ideal
- Glass/plastic (2-8 dielectric constant)
- Adhesive: Thin, high permittivity, no air bubbles
  2mil (50micron)