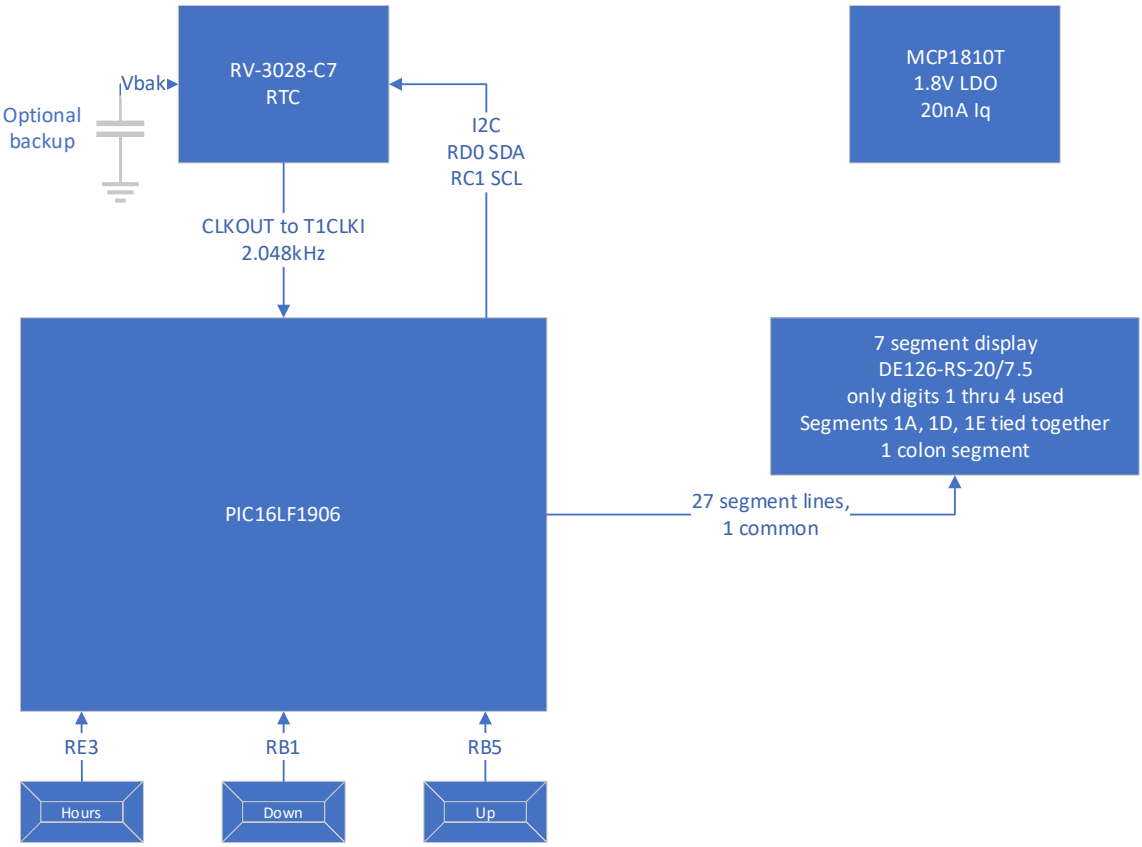
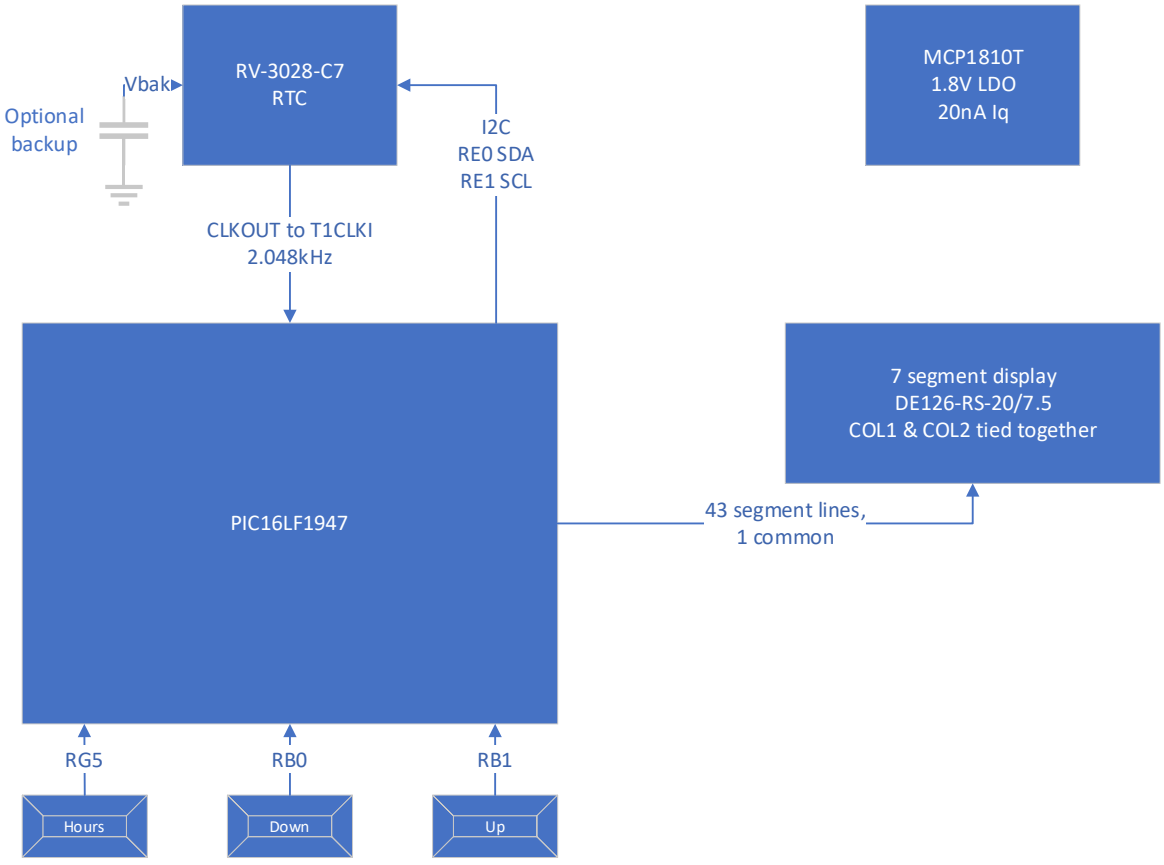
The NanoWatt Clock

Four digit:



Six digit:



NanoWatt Clock

- Aim: a six digit clock that consumes **less than 1uW**
- 2017: 5.5uW – PIC24FJ256GA410 – 3.0v @ 1.8uA
- Can we do better?

Considerations for ultra low power: a recap & update

- Base sleep current (no peripherals active) – 10s of nA
- Peripherals active in sleep – LCD, I/O, RTC/timer. Optionally WDT, BOR.
- Active power: choose lowest duty cycle and lowest uA/MHz
- Fastest wake up
- Biasing networks
- On-chip regulators – beware
    - New external LDOs available with very low Iq ~20nA
- Beware Vcap voltage when measuring current
- Static LCD
- Vbat – not always a good thing
- Board cleaning
    - eg Electrolube SWAS detergent in ultrasonic cleaner
    - Then two rinses in deionised water, dry off

Drawbacks of 2017 design

- Datasheet low-voltage MCU sleep current 60nA, measured ~500nA
    - (Datasheet DS30010089E now updated to 630nA!)
- On-chip “low power” oscillator relatively power hungry – ~170nA
- Running multiple separate divider chains in sleep at 32 kHz is power hungry (ie, both LCD and internal RTC or timer)

Hardware

- PIC16LF1907 (4-digit) or PIC16LF1947 (6-digit)
  - Base sleep current measured at around 15nA at 1.8V and 3.0V
- Run at 1.8V: substantial reduction in dynamic power consumption
- Use ultra low Iq LDO, eg, MCP1810T (20nA) or TPS7A02 (25nA)
- I2C is in software to save pins
- Buttons on pins with internal weak pullups, two on IOC pins
- Saving segment pins
  - 4-digit version: Combine 10Hour A, D, E segments, 27 seg pins total
  - 6-digit version: Combine COL1 & COL2 segments, 43 seg pins total
- Why not more modern PIC16LF19197?
  - Higher sleep current, over 100nA
- **External RTC - MicroCrystal RV-3028-C7**
  - Features:
    - 45nA @ 3.0V, operates from 1.1V to 5.5V
    - 1ppm @ 25°C, temperature compensated
    - Push-pull CLKOUT and open drain INT outputs
    - CLKOUT has programmable & compensated frequency
    - Vbackup with optional trickle charge for capacitor or battery
    - Factory programmed frequency offset calibration
    - I2C
    - POR detection
  - Gotchas:
    - I2C expensive in power terms
    - Wait for RTC to start up on POR and “refresh” RAM from EEPROM
    - Refresh runs at midnight every day
    - INT and CLKOUT are not mutually exclusive outputs
    - Be careful to use low leakage caps on Vbackup
    - Vbackup switching uses additional 45nA
- RTC’s CLKOUT 2.048kHz clocks MCU’s LCD and timer via T1CLKI
- LCD runs at 16Hz

Software

- Use on-chip Timer1 clocked by external RTC to wake MCU
- Time is maintained by MCU because I2C expensive in power terms
    - Except after MCU POR, when time is read from RTC if it's still valid
- 4-digit version:
    - 2,000,000:1 duty cycle: 30us in every 60s, ~250pA aggregate CPU power
- 6-digit version:
    - 16,667:1 duty cycle: 30us every 0.5s, ~30nA aggregate CPU power
- Low power delay
- Software I2C
- MCU polls RTC on reset every 50ms to wait for refresh completion
- MCU resets to compile build time instead of RTC time if:
    - RTC POR event occurred
    - MCU has just been programmed (uses on-cip HEF or EEPROM)
    - Debug build
- Using MCLR pin as digital input, so LVP disabled

Results

- Measured at LDO's Vin (3.0V)
    - 4-digit: 403nA average with 19 segments on (1209nW)
    - 6-digit: 514nA average with 29 segments on (1542nW)
- Measured at LDO's Vout (1.8V)
    - 4-digit: 385nA average with 19 segments on (693nW)
    - 6-digit: 498nA average with 29 segments on (896nW)
- Add 45nA for RTC Vbackup switching

Possible improvements

- Sleep during low power delay
- External 1M pull ups instead of internal pull ups