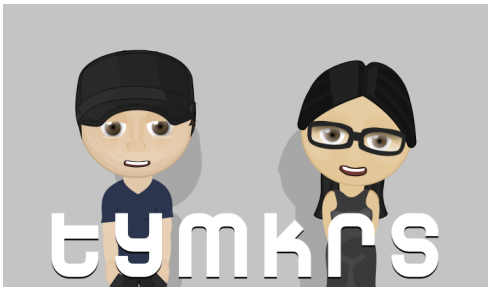
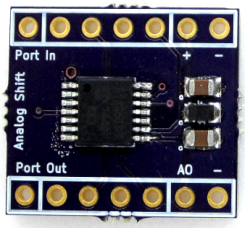
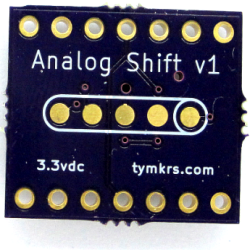


The Toymakers @ tymkrs.com
Questions? Please contact us:
feedback@tymkrs.com

DATASHEET





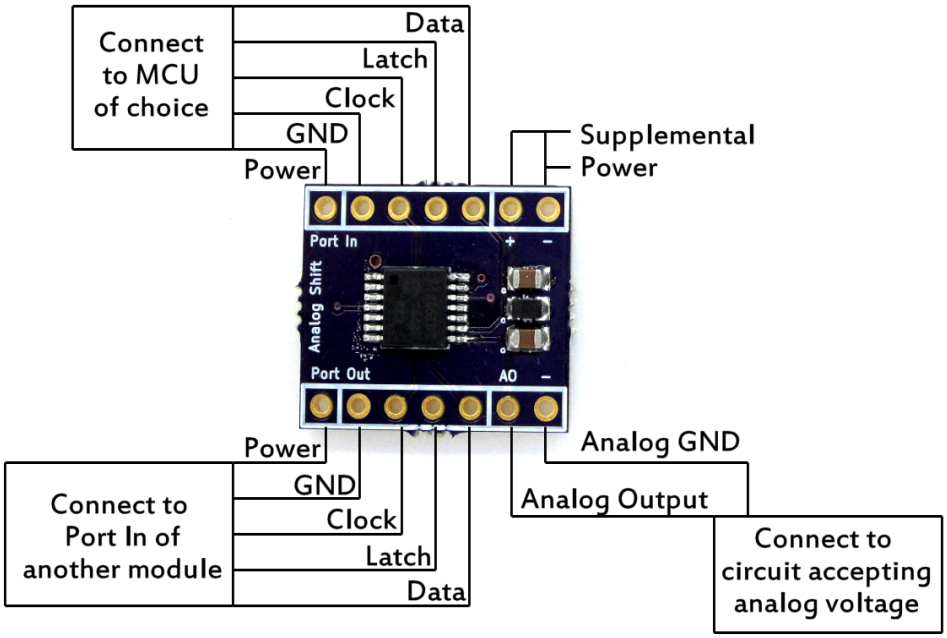
Analog Shift

0 – 3.3V DAC

The Analog Shift takes any 7-bit digital value and converts it into a voltage between 0V and 3.3V while also acting like a shift register to shift those bits through.

- Kit Type: SMT kit with minimal throughhole soldering
- Function: Takes a 7-bit digital value between 0000000 and 1111111 and converts it into a voltage between 0V and 3.3V. It is capable of shifting the bits through the module to another.
- Uses 3 pins, VCC, and GND on the MCU to send serial data to the shift register

Board Pins



Contents of the AND8 Module:

- AND8 printed circuit board (18.39 x 15.85 x 1.60mm) – reflowed for you already
- 2 – 1x8 male headers

Electrical Components:

Reference	Quantity	Type	Value
--	1	PIC Microchip, 14 pin TSSOP	PIC16F1705
C1	1	Capacitor, 16V	0.1 uF
C2	1	Capacitor, 25V	10.0 uF
C3	1	Capacitor, 16V	0.1 uF

PIC16F1705 Operating Conditions
Datasheet: http://ww1.microchip.com/downloads/en/DeviceDoc/40001729A.pdf

Parameter	Ratings	Unit
Supply Voltage	3.3	V
Operating Temperature	-40 to +85	ºC
Speed	32	Hz
Drive current	70mA @ 3.3V	---

Tools and material required for assembly (not included with the kit):

- Soldering iron and Solder

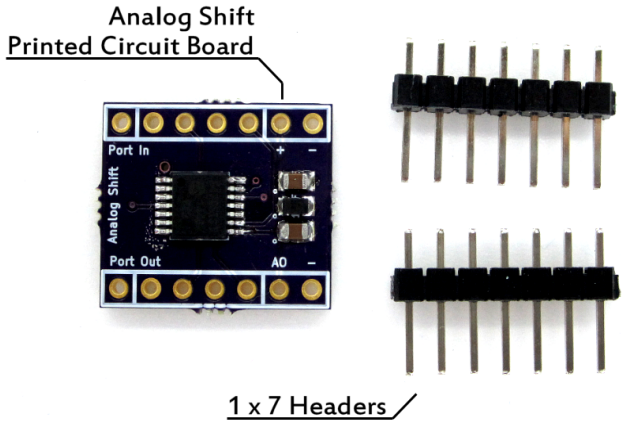
User provided items required for intended function:

- Microcontroller of choice
- Post module circuit utilizing analog voltage

Additional physical/electrical specifications:

- Printed Circuit Board size: 0.72 x 0.62 x 0.063" (18.39 x 15.85 x 1.60mm)
- PCB thickness: 0.063" (1.60mm), not including any components
- PCB thickness: 0.433" (11.0 mm), max height with headers.
- Headers are breadboard friendly.

Additional Picture:



Kit Parts

Use Instructions

- **Method of use:** Control of the Analog Shift by the microcontroller of choice requires at minimum Clock, Latch, Serial, and GND. Power and GND can come from the microcontroller or by way of the supplemental power header.
- **Requirements:** GND of the microcontroller should be connected to GND of the Analog Shift module as well as the GND of the resultant circuit used by the Analog Output. The voltage between the microcontroller and the Analog Shift must also be the same. Ie. If your microcontroller runs off of 3.3V, your board needs to be powered with 3.3V.
- **Daisy-chaining**: Multiple Analog Shifts can be connected to each other. The Port Out of one Analog Shift can be connected to the Port In of another Analog Shift. The microcontroller sends the serial data to all chained Analog Shifts to simultaneously give x number of analog voltages. The Port Out portion of the PCB passes through the digital data coming into the Port In section from the Microcontroller.
- When coding the analog shift, treat it as if it were a shift register. The resultant DAC voltage is based on the lower 7 bits.
    - The resultant analog voltage is based on Bits 0 to 6 assuming we're inputting an 8 bit value 0-7.

Example Code

```
{{
┌──────────────────────────────────────────┐
File: AnalogShiftDemo.spin
Version: 1.0
Copyright (c) 2015 Tymkrs
See end of file for terms of use.

Author: Whisker
└──────────────────────────────────────────┘
}}

{
HISTORY:
  This object is made as an example for using the Analog Shift kit from
http://tymkrs.com/

USAGE:

  • Connect Analog Shift 'Port In' pins Clock (Port In header pin 3),
    Latch (Port In header pin 4),
    and Data (Port In header pin 5) to
    Propeller pins Clock_Pin, Latch_Pin, and Data_Pin respectively

  • Connect Analog Shift 'Port In' pins Vcc (Port In header pin 1),
    and Vss (Port In header pin 2) to +3.3v and GND respectively

}
Con
  _clkmode = xtal1 + pll16x
  _xinfreq = 5_000_000

  'Define which Propeller pins are connected to the Analog Shift
  Clock_Pin     = 0
  Latch_Pin     = 1
  Data_Pin      = 2

Pub Main | Index, Level

  'Set 'Port In' ports to be output pins
```

```
  dira[Clock_Pin] := %1
  dira[Latch_Pin] := %1
  dira[Data_Pin] := %1

  'Loop demo forever
  repeat

    'Cycle the value of 'Level' from 0 through 127
    repeat Level from 0 to 127

      'Put current value of Level into the dirb register
      dirb := Level

      'Cycle through all 8 bits of our Level value
      repeat Index from 0 to 7

        'Set Data output pin to this bit
        outa[Data_Pin] := dirb[Index]

        'Clock this bit out to the Analog Shift
        outa[Clock_Pin] := 1
        outa[Clock_Pin] := 0

      'Apply the bits stored in the Analog Shift register to the Analog Output
(AO pin)
      outa[Latch_Pin] := 1
      outa[Latch_Pin] := 0

      'Pause for half a second before looping
      waitcnt((clkfreq / 2) + cnt)

{{
```