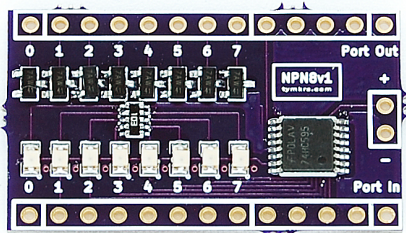**The Toymakers @ tymkrs.com**
Questions? Please contact us:
feedback@tymkrs.com

DATASHEET

NPN8 Module

The NPN8 kit allows you to control 8 NPN BJT transistors simultaneously via serial-to-parallel shift register to drive 8 end devices/components.

- Kit Type: SMT kit with minimal throughhole soldering
- Function: Drives 8 end devices via use of serial-to-parallel shift register
- Uses 3 pins, Vcc, and GND on the MCU to send serial data to the shift register
- The module is daisy-chainable and extra power rails are available to supplement extra power needs.

KIT CONTENTS
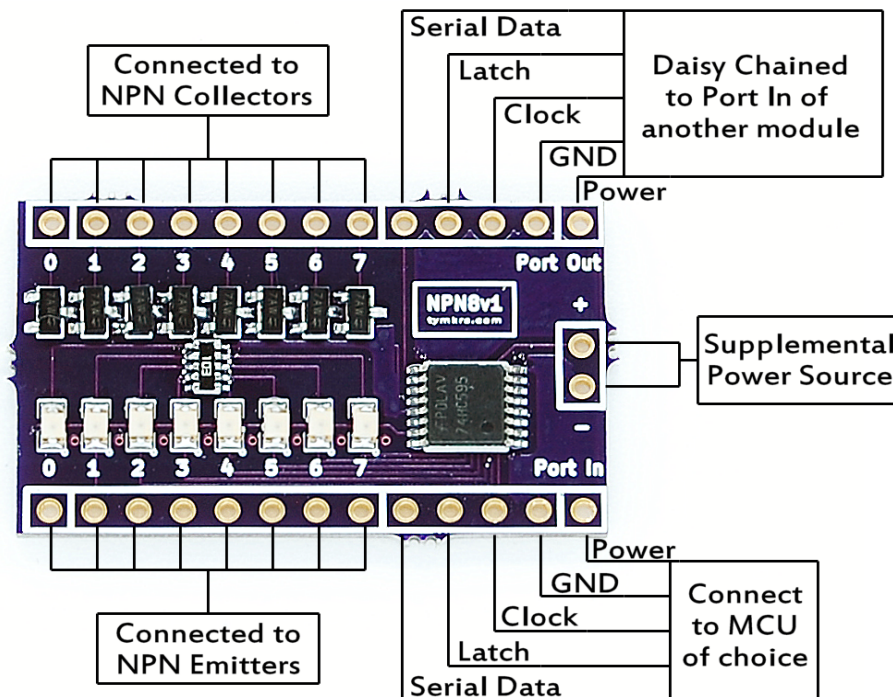
Contents of the NPN8 Module Kit:

- NPN8 printed circuit board (34.14 x 20.93 x 1.60mm) – reflowed for you already
- 2 – 1x12 male headers

Electrical Components:

Reference	Quantity	Type	Value
C1 - C8	8	NPN BJT Transistor SOT-23, 40V	2N3904
D1 – D8 (optional)	8	LED, 0805, Vf = 1.8V, 20mA	Hyper red LED
-- (optional)	1	Resistor Array, 1206	240 ohm
--	1	Shift Register, 16-TSSOP	74HC595

75HC595 Shift Register Maximal Operating Conditions
Datasheet: http://www.nxp.com/documents/data_sheet/74HC_HCT595.pdf

Parameter	Maximal Ratings	Unit
Supply Voltage	-0.5 – +7.0	V
Operating Temperature	-40 to +125	ºC
Output Current (Qn)	+/- 35 per pin	mA
Supply Current	70	mA

2N3904 BJT NPN Transistor Maximal Operating Conditions
Datasheet: http://www.nxp.com/documents/data_sheet/MMBT3904.pdf

Parameter	Max Ratings	Unit
Collector-emitter voltage	40	V
Operating Temperature	-65 to +150	ºC
Collector Current (DC)	200	mA
Peak Collector Current	200	mA
Peak Base Current	100	mA
Total Power Dissipation	250	mW

Recommended Operating Conditions

Parameter	Ratings	Unit
Supply Voltage	3.3 – 5.0	V
Ambient Temperature	25	ºC

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

- Soldering iron
- Solder

User provided items required for intended function:

- End device (such as LEDs, triggers, motors, etc)

Additional physical/electrical specifications:

- Printed Circuit Board size: 1.34 x 0.82 x 0.063" (34.14 x 20.93 x 1.60mm)
- PCB thickness: 0.063" (1.60 mm), not including any components
- PCB thickness: 0.433" (11.0 mm), max height with headers.
- Headers are breadboard friendly.

Use Instructions
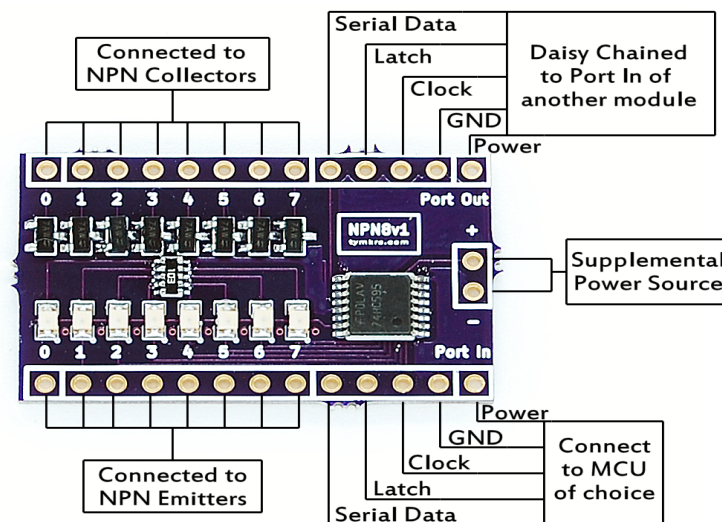
- **Method of use:** Control of the NPN8 by the microcontroller of choice requires at minimum Clock, Latch, Serial, and GND. Power and GND can come from the microcontroller or by way of the supplemental power header.
- **Requirements:** GND of the microcontroller should be connected to GND of the NPN8 module. The voltage between the microcontroller and the NPN8 must also be the same. Ie. If your microcontroller runs off of 3.3V, your board needs to be powered with 3.3V. This is due to the comparator in the shift register which determines what a 1 and what a 0 is.
- **Daisy-chaining**: Multiple NPN8s can be connected to each other. The Port Out of one NPN8 can be connected to the Port In of another NPN8. The microcontroller would still send the serial data to all chained NPN8s.

Example Code

```
File: NPN8_Demo_1.spin
Version: 1.0
Copyright (c) 2013 Tymkrs
See end of file for terms of use.
'This code shows the binary value of what you type out on a keyboard in LEDs
Author: Whisker
```

```
CON

  _clkmode = xtal1 + pll16x
  _xinfreq = 5_000_000
  CLS = 16
  CR = 13
  Clock = 2 'Set the MCU pins that will control the NPN8 module
  Latch = 1
  Serial = 0


OBJ

  keys: "Keyboard"
  debug: "FullDuplexSerial"


PUB KeyboardDisplay | Index

  keys.start(26,27)
  'debug.start(31,30,0,57600)
  dira[Latch]            := 1 'set MCU pins as outputs
  dira[Clock]            := 1
  dira[Serial]           := 1


  waitcnt(clkfreq + cnt)

  'debug.str(string(CLS, "Type characters on", CR, "the PS/2 keyboard:", CR, CR))

  repeat
    if keys.gotKey 'This monitors the keyboard to see what key you have pressed
      dirb := keys.getKey
      debug.tx(keys.gotKey)

    repeat Index from 0 to 7
      'Set the state of LED Serial Pin for this LED to the value stored in its slot of
the SwitchState array
      outa[Serial] := dirb[Index]
      'Pull the LED Clock Pin high then low to write this LED's state into the 595's
register
      outa[Clock] := 1
      outa[Clock] := 0
      'Pull the LED Latch Pin high then low to apply the contents of the 595's register
to the 595's output pins (LEDs)
    outa[Latch] := 1
    outa[Latch] := 0
```