AVR Freaks

DESIGN NOTE #044

AUTHOR: LUKE SANGALLI

KEYWORDS: XXXXXXXXX

Using Trinary Encoding to Reduce Pin Wastage

Introduction

Often enough a device will need to read some dip-switches to preset a range of function or settings. Trinary coded dip-switches increase dramatically the number of options available from a fixed number of I/O pins used to read the dip-switch settings. The AVR microcontroller can achieve this relatively simply by using the programmable pull-up feature on its I/O Ports without wasting an additional port bit.

Overview

I/O pins on most microcontrollers are binary input devices, i.e., they will recognise only two states HIGH/LOW so that for three inputs the maximum number of selectable combinations is $2^3 = 8$. If the same number of inputs were trinary encoded the total number of selectable combinations would now be $3^3 = 27$, a dramatic increase indeed. Table 1 below lists the respective number of selectable combination in both binary and trinary.

Table 1. Binary and Trinary Combinations

Bits	Binary Combinations	Trinary Combinations
1	2	3
2	4	9
3	8	27
4	16	81
5	32	243
6	64	729
7	128	2187
8	256	6561

One other advantage of trinary encoding of dip-switches is that often the functions are separated in groups (e.g., in a UART, Baud Rate, parity, and number of stop bits are three separate entities and are encoded separately).

When one of these entities is not a disable/enable binary function extra bits are needed to encode wasted states.

In the UART example above:

For Baud Rates of 300,1200,2400,4800,9600,19200,38400,57600,115200

Parity None, Odd or Even

Stop Bits 1, 1.5, 2

See Table 2 below:

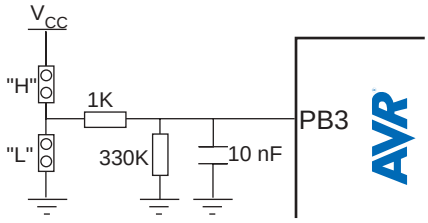
Table 2. Function Settings and Encoding

Function	Setting	Binary Encoded	Trinary Encoded
Baud Rate	300	L L L L o o o o	L L o o
	1200	L L L H o o o o	L H o o
	2400	L L H L o o o o	L Z o o
	4800	L L H H o o o o	H L o o
	9600	L H L L o o o o	H H o o
	19200	L H L H o o o o	H Z o o
	38400	L H H L o o o o	Z L o o
	57600	L H H H o o o o	Z H o o
	115200	H L L L o o o o	Z Z o o
Parity	None	o o o o L L o o	o o L o
	Odd	o o o o L H o o	o o H o
	Even	o o o o H L o o	o o Z o
Stop Bits	1	o o o o o o L L	o o o L
	1.5	o o o o o o L H	o o o H
	2	o o o o o o H L	o o o Z

A total of eight binary encoded bits would be required compared with four trinary encoded bits.

The way we achieve this is to use a detectable weakly grounded almost Hi -impedance state Z. See Figure 1 below.

Figure 1. Trinary Encoded Pin Schematic



To set the state to LOW you jumper "L"
To set the state to HIGH you jumper "H"
To set the state to HI-Z you leave both "L" and "H" unjumpered

In order to detect what state PB3 is in you can follow the following procedure:

- Set PB3 to input
- Turn the AVR pull-ups on PORT B off
- Read PB3
- If PB3 is 1 then the set state of PB3 is HIGH
- If PB3 is 0 then turn the pull-ups on PORT B on
- Read PB3 again
- If PB3 is still 0 then the set state of PB3 is LOW
- If PB3 is now 1 then the set state of PB3 is HI-Z

The reason this works is thanks to the AVR's internal pull-ups (approximately 35 - 120 KΩ) which in this case allow you to detect a weakly grounded input pin as well as the usual high and low states making it three states.

When PB3 is pulled high via the 1K resistor whether the pull-ups are on or off you will always detect a high state.

When PB3 is pulled low via the 1K resistor the internal pull-up with a minimum resistance of 35K cannot possibly lift the voltage past V_{CC} x 1/36 and therefore will always detect a low state.

If both the "L" and "H" jumpers are left open PB3 is weakly held low by 330K resistor and when the pull-ups are enabled will allow PB3 to climb sufficiently to detect it as a high thus giving us a third state.

The capacitor to ground reduces spurious noise pick-up especially if your tracks are long or you've got high currents being switched nearby or on the same ground.