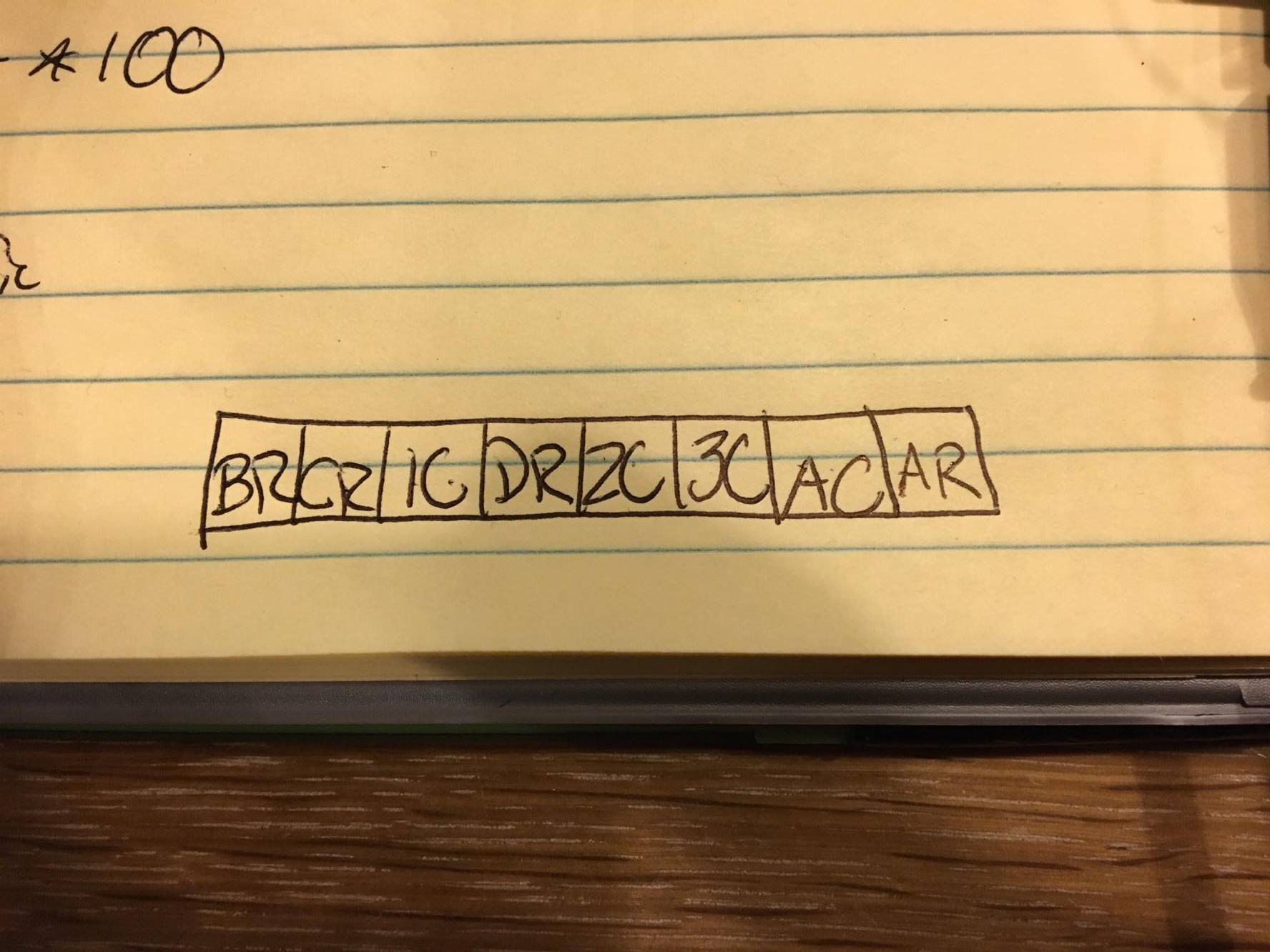
1. The TCNT1 register is used to generate the low frequencies of the DTMF generator (the columns). The OCR1A/B register is used as an output/compare for the timer to know when to start and stop the pulses. TIMSK1 is used to configure the interrupts for Timer 1. TCCR1A/B is used to enable the CTC mode.
2. Minimum and maximum frequency respectively of Timer 1 are 122Hz and 8,000,000Hz. For Timer 0, it is 122Hz and 31,372Hz.
3. 1 clock tick = 6.25x10^(-5)ms.
4. Minimum and maximum frequency respectively of Timer 1 are 122Hz and 8,000,000Hz. For Timer 0, it is 122Hz and 31,372Hz.
5. 

The first character in this diagram is the key that’s being considered, and then the second character is either an R for row, or a C for column of the first character.

1. We used the Timer2 Compare Output register to set “stallTimes” and had that updated every time the interrupt was made. To scan for keypresses, we wired up our keys to be high on columns, and then rows would be ground. We would then alternate columns by making one low and the rest high. Using pull-up resistors, we ensured the voltage would go to ground when the switch became active. Then using the PortD GPIO pins, we set them to input and would check to see if one was going low.
2. We would use pull-up resistors to ensure current was flowing. The keypad acted as a switch for the column and row so we could determine which key was being pressed. This is a pulldown switch. If it was a pullup switch, we would only need to change the output to low on PortB registers and then alternate one column each clock tick as high to scan for key presses.
3. We didn’t use a prescaler, but it would be useful in order to be able to set clock ticks more easily and achieve lower frequencies.
4. CTC mode is easier to deal with, but you have to consistently change the frequencies. Using interrupts, its easy to change the frequencies you need to output, but deals with overflows and other interrupts which may have more bugs.