

Proyecto de ejemplo de fuentes de restablecimiento de AVR®


Objetivo

Este proyecto pasa por varias condiciones de reinicio diferentes: reinicio *de encendido (POR)*, reinicio *de apagado (BOR)* y tiempo de espera *del temporizador de vigilancia (WDT)*, y muestra cómo funciona cada uno en una **placa Xplained** de 328 PB. Se muestra que algunos circuitos externos producen una entrada de voltaje variable, pero también funcionará una fuente de alimentación ajustable.


Para obtener más detalles sobre las **fuentes de restablecimiento de AVR®**, visite [Descripción general de las fuentes de restablecimiento de AVR.](#)

Materiales

Herramientas de hardware (opcional)

Herramienta	Sobre	Compra
 Mini kit de evaluación ATmega328PB Xplained		

Herramientas de software

Herramienta	Sobre	Instaladores			Instrucciones de instalación
		ventanas	linux	Mac OS X	
 Entorno de desarrollo integrado Atmel® Studio					

Archivos de ejercicios

Expediente	Descargar			Instrucciones de instalación
	ventanas	linux	Mac OS X	
 Archivos de proyecto y fuente				

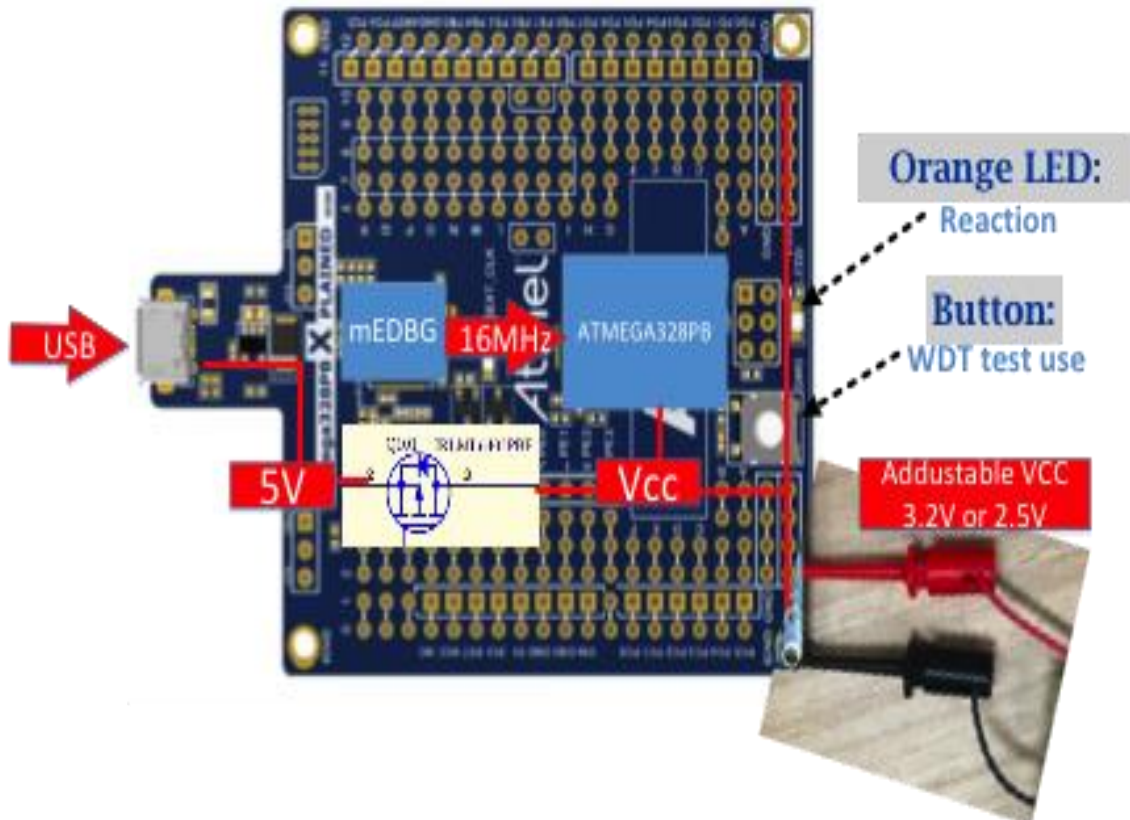
Archivos adicionales

archivos



Guía del usuario de la miniplaca Xplained 328PB

Diagrama de conexión

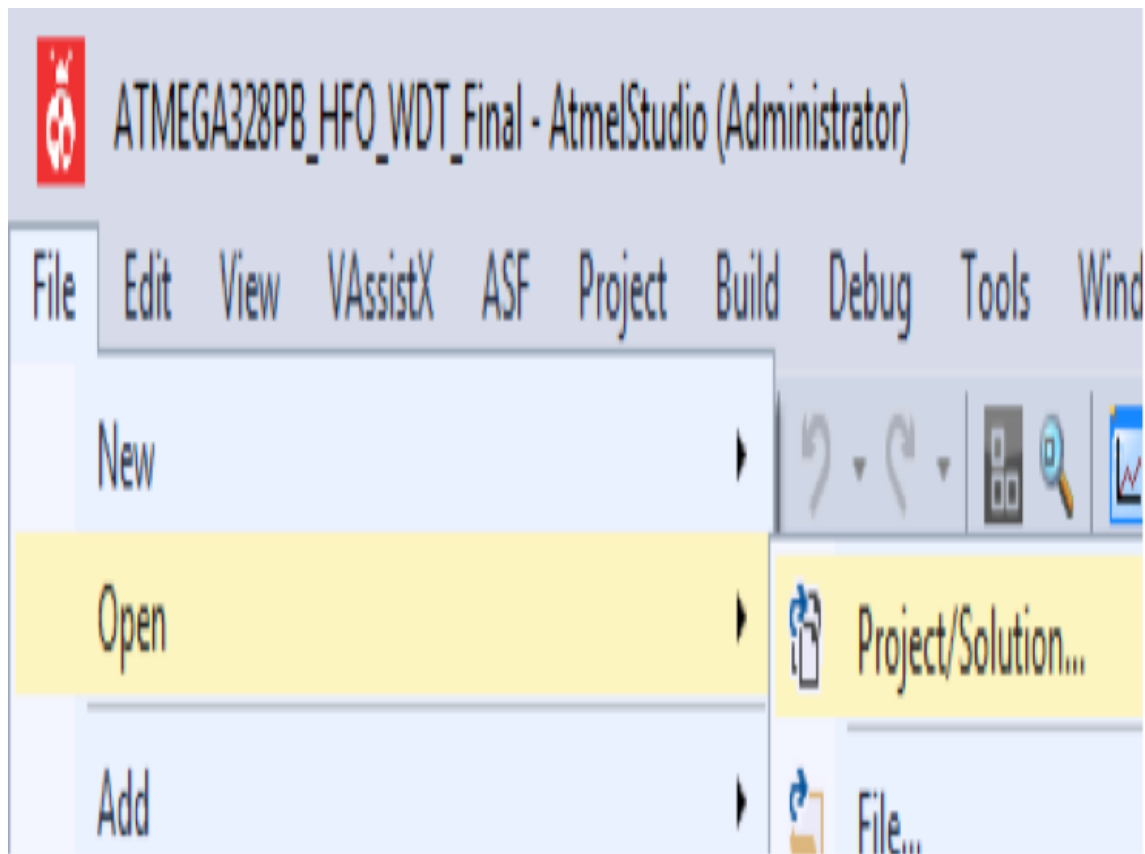


Procedimiento

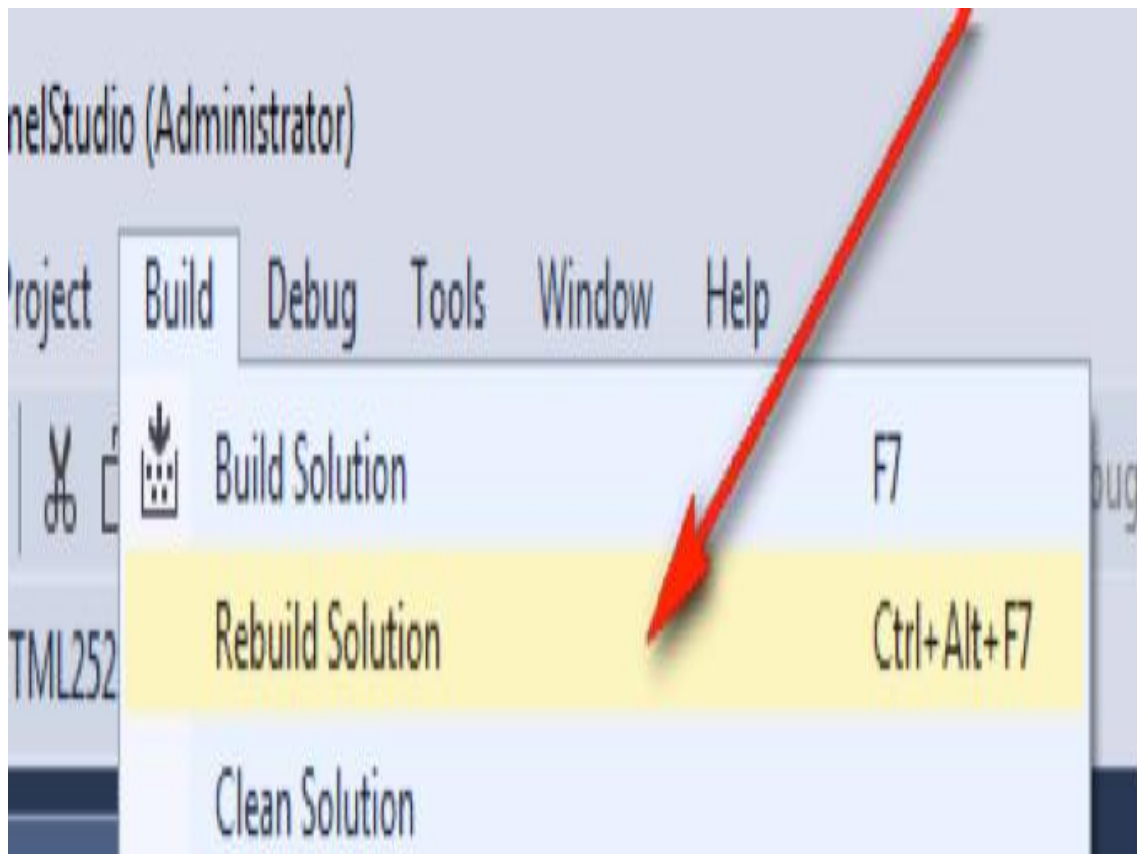
1

Tarea 1: abrir y compilar el proyecto

- Descargue los archivos fuente del proyecto de la sección anterior Archivos de ejercicios y descompríalos en su computadora.
- Abrir Atmel Studio 7
- Seleccione **Archivo > Abrir > Proyecto/Solución**



- Seleccione `ATMEGA328PB_HFO_WDT_Final.atsIn` de los archivos de proyecto descargados.
- En el menú Generar, seleccione 'Reconstruir solución'.

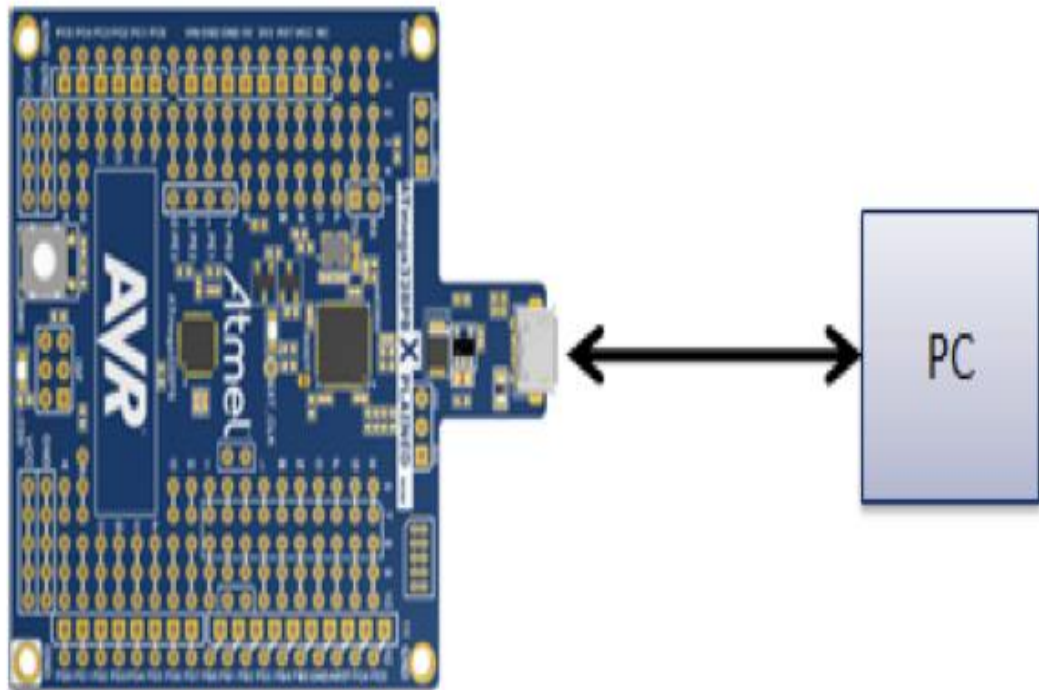


Seleccione 'GCC C Executable Project' y asígnele el nombre `Project1` . Elija una ubicación para guardar el proyecto en su computadora.

2

Tarea 2: preparación de la placa

Asegúrese de que el cable USB esté conectado entre la placa y la PC.



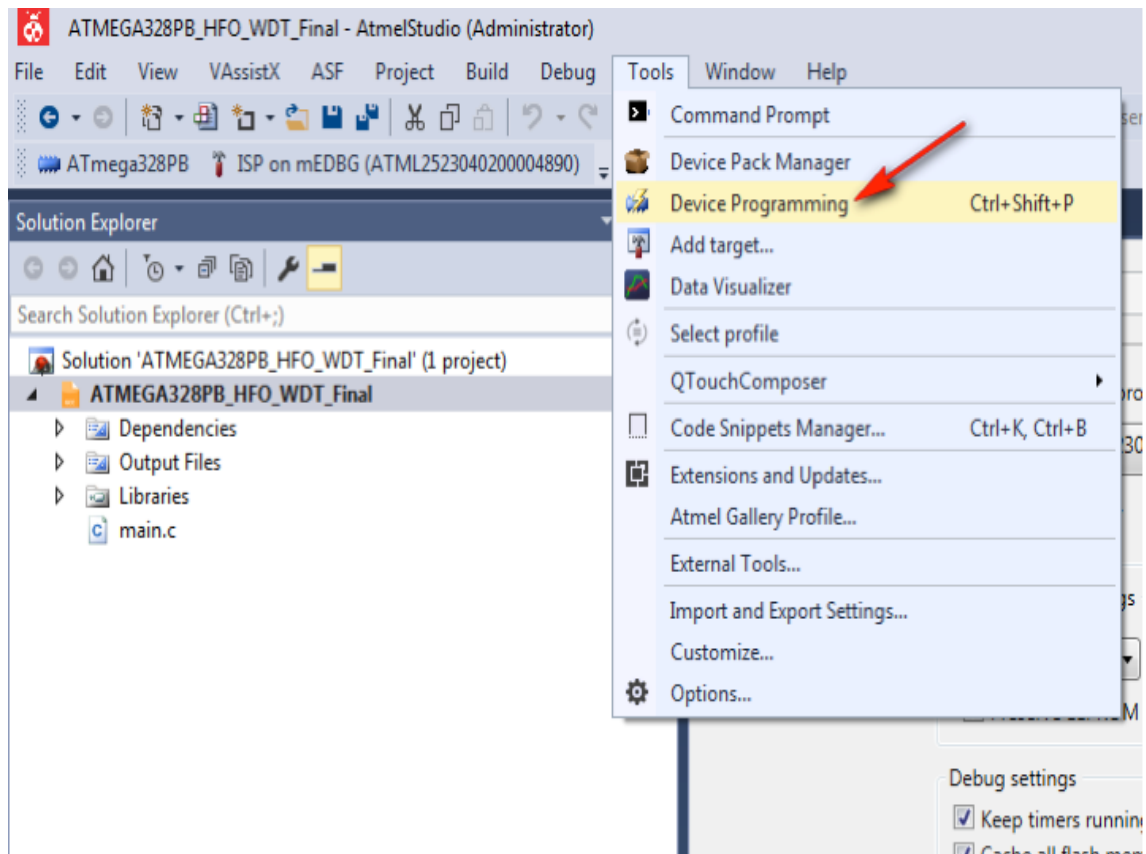
3

Tarea 3 - Configuración del programador

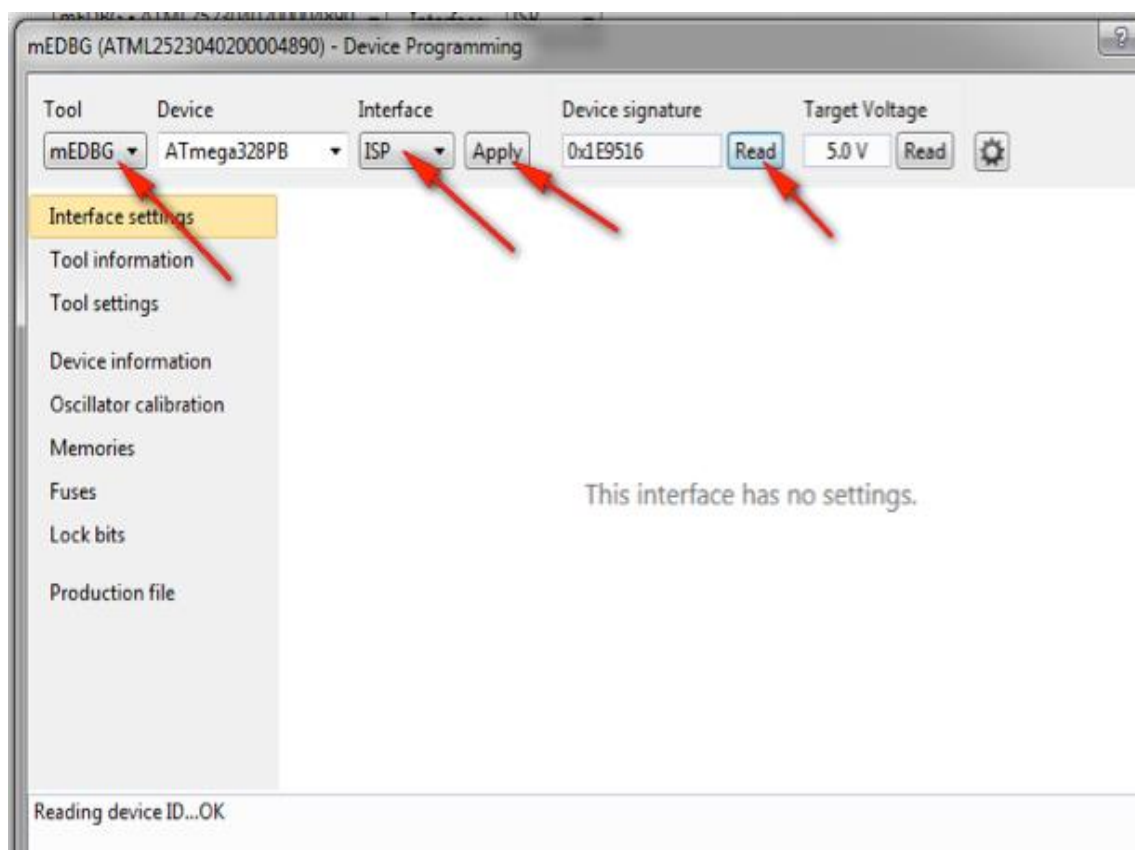
Si debugWire está habilitado, desactívalo.

[Deshabilitar debugWire \(DWEN\) ►](#)

- Haga clic en **Herramientas > Programación de dispositivos** :



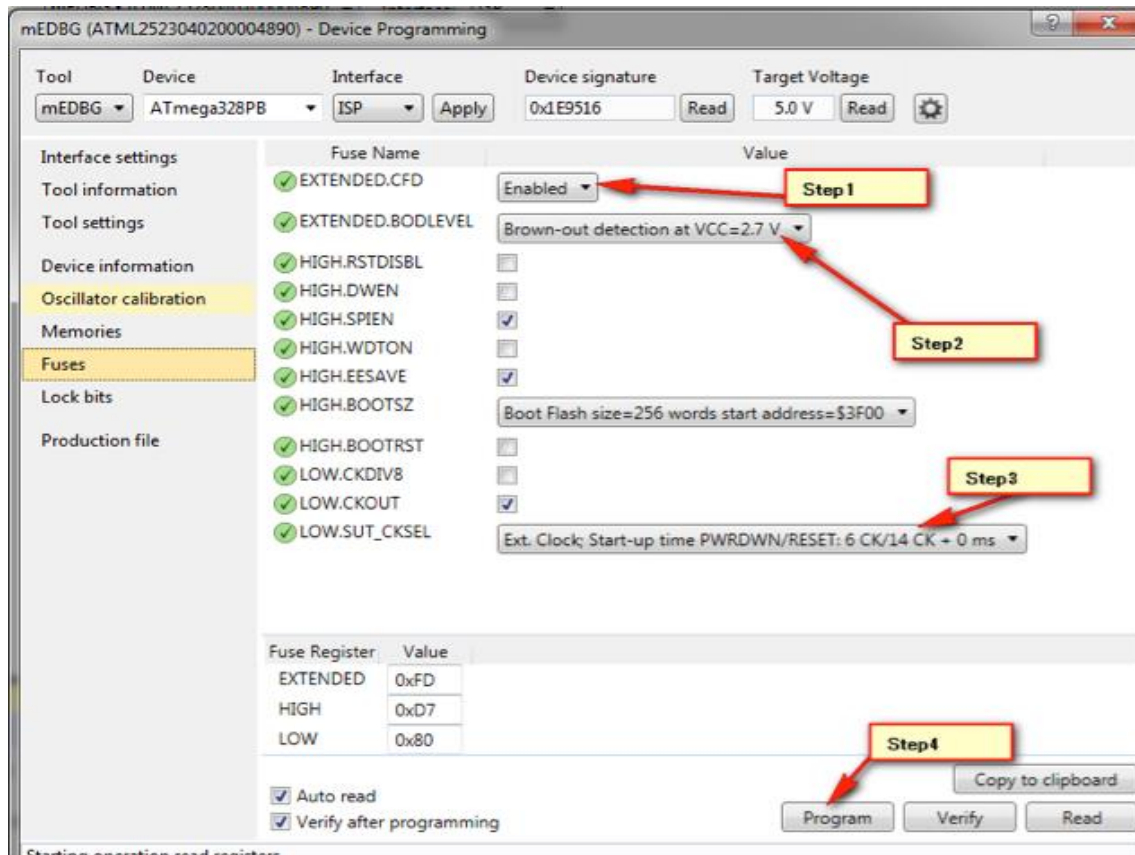
- Haga clic en los botones que están marcados como flechas rojas en la captura de pantalla.



4

Tarea 4 - Programación de los bits de fusible

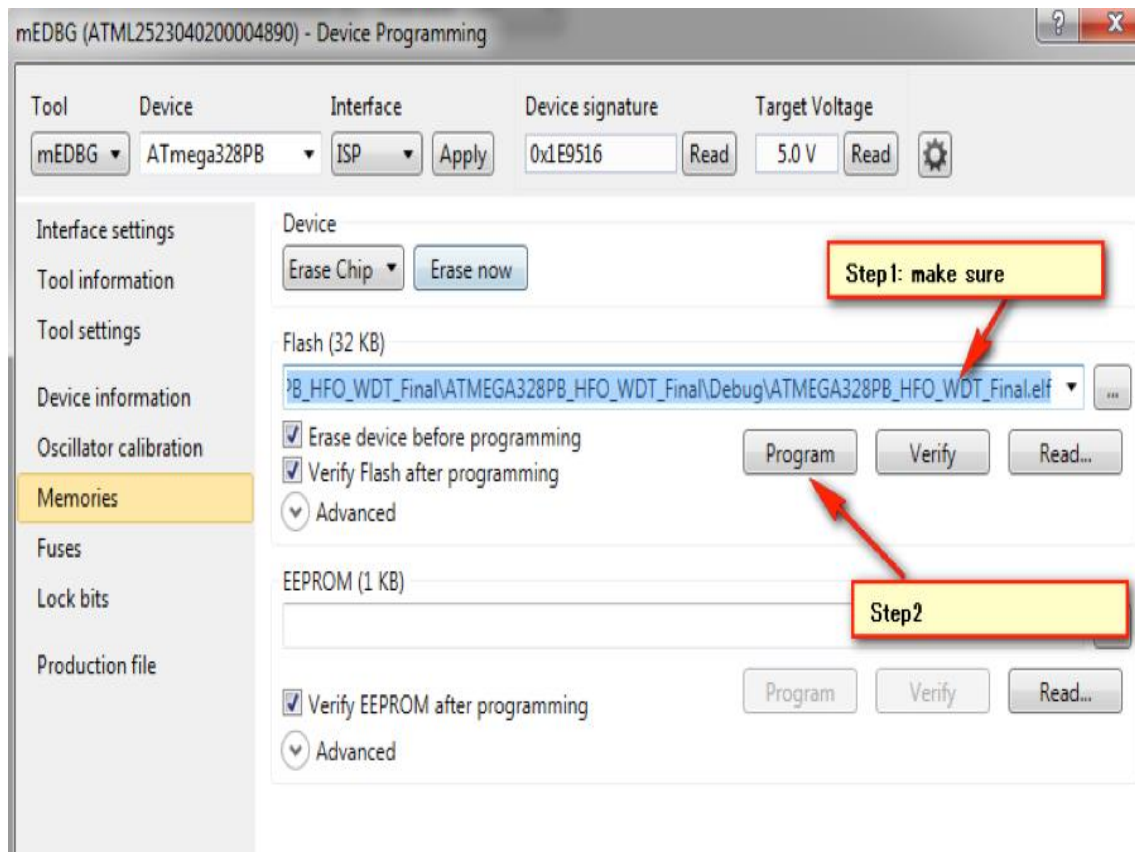
- Seleccione las opciones de 'Fusibles' en la barra de opciones de la izquierda y establezca los bits como se muestra en la figura, lo que habilitará CFD, umbral de DBO como 2,7 V y reloj externo.



5

Tarea 5 - Programación de dispositivos

Seleccione las opciones de 'Memorias' en la barra de opciones de la izquierda y termine los pasos como se muestra en la figura, que programa el archivo binario compilado en ATMEGA328PB.



6

Tarea 6 - Código del proyecto

Aquí está el código completo al que hacemos referencia. También puedes descargarlo en la Sección de Ejercicios.

```
/*
 * ATMEGA328PB_HFO_WDT_Final.c
 *
 * Created: 2017/03/08 17:15:35
 * Author : A17582
 */

#define F_CPU 8000000UL

#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/wdt.h>
#include <util/delay.h>
```



```

//initialize watchdog
void WDT_Init(void)
{
    //disable interrupts
    cli();

    WDTCSR = (1<<WDCE)|(1<<WDE); // Enable configurat
ion change.

    WDTCSR = (1<<WDIE)| // Enable Watchdog
Interrupt Mode.

    (1<<WDCE)|(1<<WDE)| // Enable Watchdog
System Reset Mode if unintentionally enabled.

    (0<<WDP3)|(1<<WDP2)|(1<<WDP1)|(1<<WDPO); // Set Watchdog Ti
meout period to 4.0 sec.

    //Enable global interrupts
    sei();
}

//Watchdog timeout ISR
ISR(WDT_vect)
{
    //Burst of 0.1Hz pulses
    for (uint8_t i=0;i<4;i++)
    {
        //LED OFF
        PORTB &= ~(1 << PINB5); // Set PORTB5 Low
        _delay_ms(80);
        //LED ON
        PORTB |= (1 << PINB5); // Set PORTB5 On
        _delay_ms(20);
    }
}

#define BORFbit 2

```

```

#define PORFbit 0

int main(void)
{
    unsigned char i;
    DDRB |= (1 << PINB5);           // Set PORTB5 as output ,
    DDRB &= ~(1<<PINB7);           //Set PORTB7 as input

    if(MCUSR & 1 ) {
        MCUSR=0;
        for ( i=0;i<4;i++)
        {
            //LED OFF
            PORTB &= ~(1 << PINB5);           // Set PORTB5 Low
            _delay_ms(300);
            //LED ON
            PORTB |= (1 << PINB5);           // Set PORTB5 On
            _delay_ms(300);
        }
    }
    else if(MCUSR & 4) {
        MCUSR=0;
        for (i=0;i<8;i++)
        {
            //LED OFF
            PORTB &= ~(1 << PINB5);           // Set PORTB5 Low
            _delay_ms(100);
            //LED ON
            PORTB |= (1 << PINB5);           // Set PORTB5 On
            _delay_ms(100);
        }
    }
}

```

```

else if(MCUSR & 8) {
    MCUSR=0;
    WDT_Init();
    for (i=0;i<8;i++)
    {
        wdt_reset();
        //LED OFF
        PORTB &= ~(1 << PINB5);          // Set PORTB5 Low
        _delay_ms(20);
        //LED ON
        PORTB |= (1 << PINB5);           // Set PORTB5 On
        _delay_ms(80);
    }
    MCUSR=0;
}

//initialize watchdog
WDT_Init();

//delay to detect reset
//_delay_ms(500);

while(1) {

    PORTB |= (1 << PINB5);                // Set PORTB5 high
    _delay_ms(250);
    _delay_ms(250);
    _delay_ms(250);
    _delay_ms(250);

    PORTB &= ~(1 << PINB5);                // Set PORTB5 Low
    _delay_ms(250);
    _delay_ms(250);

```

```

    _delay_ms(250);
    _delay_ms(250);

    if ((PINB & 1 << PINB7) == 0) {
        PORTB |= (1 << PINB5);           // Set PORTB5 high
        _delay_ms(500);
        _delay_ms(500);
        _delay_ms(500);
        _delay_ms(500);
        _delay_ms(500);
        _delay_ms(500);
        _delay_ms(500);
        _delay_ms(500);

    }
    wdt_reset();
}

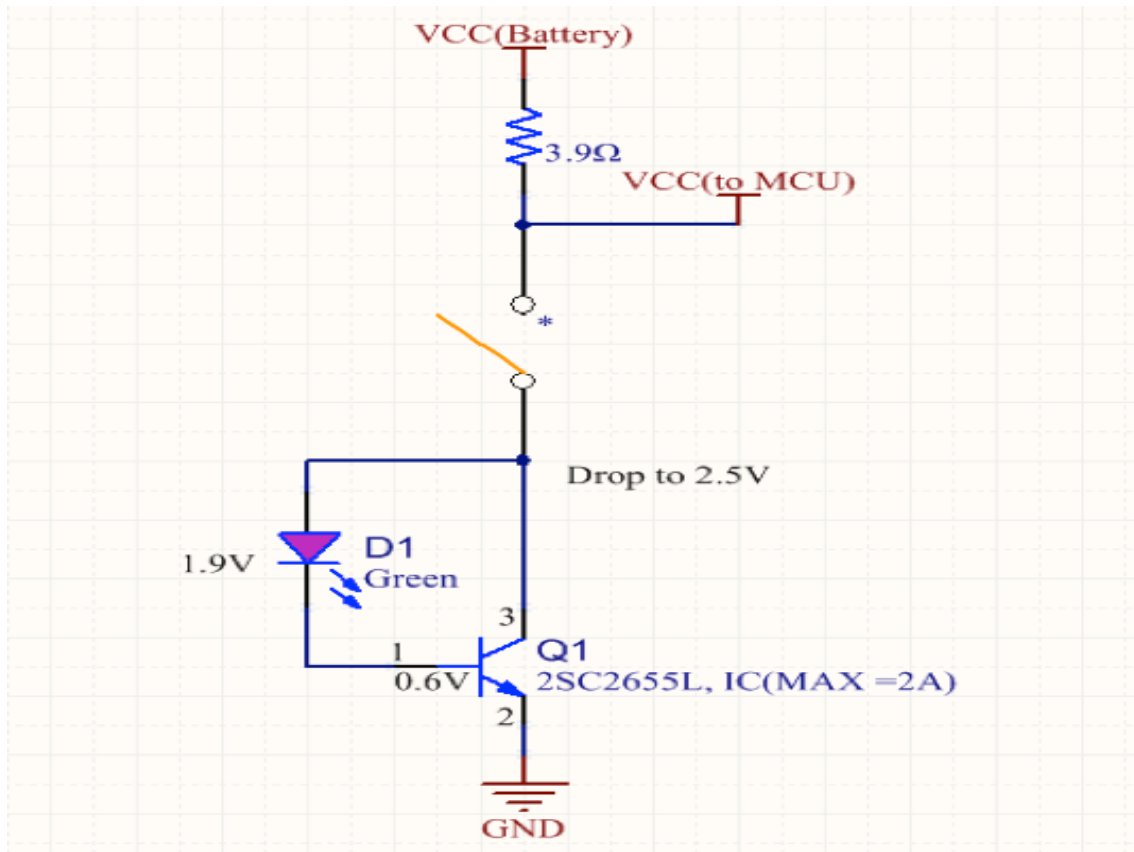
}

```

7

Tarea 7: reinicio del circuito de prueba

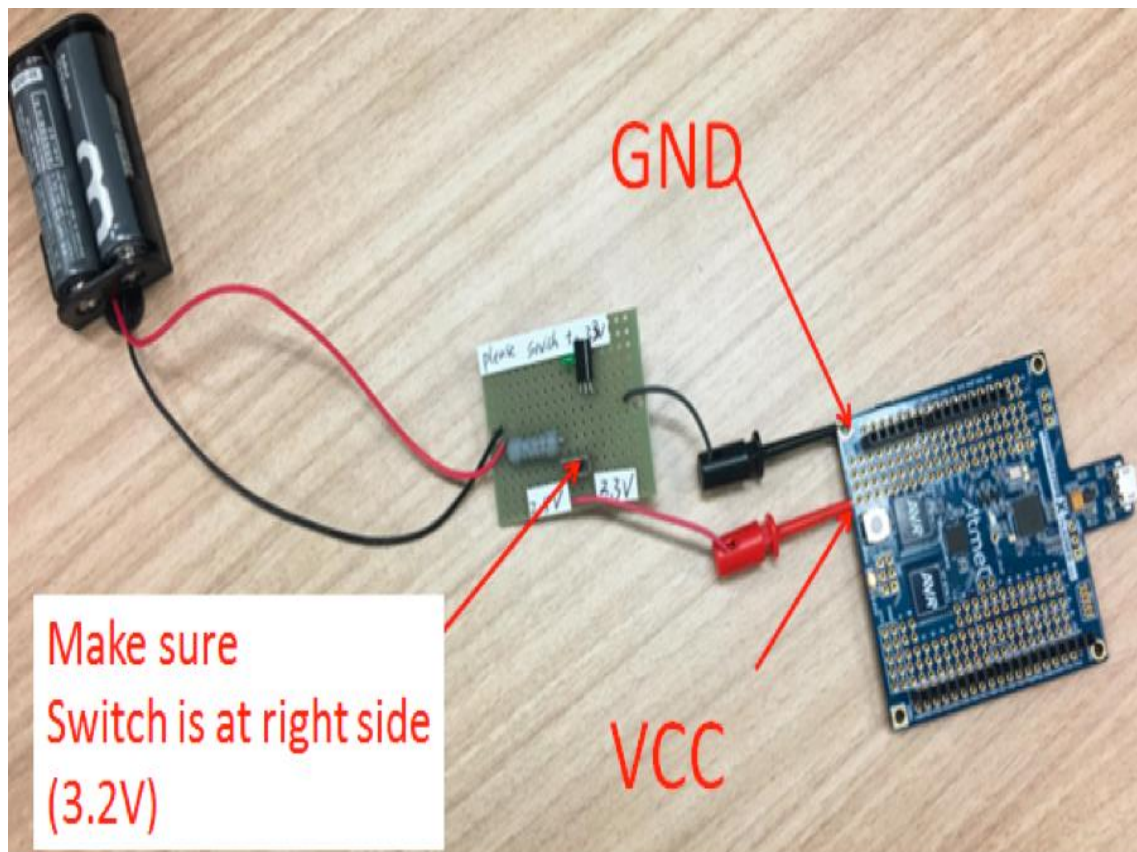
- Construya el circuito de prueba de reinicio que se muestra en el esquema.



7

Tarea 8 - Prueba POR

- Saque el cable USB
- Asegúrese de que el interruptor del cable VCC ajustable externo esté en el lado derecho (3,2 V)
- Sujete el cable VCC ajustable externo a la placa



- Resultado: el LED naranja responderá parpadeando (0,3 ms, cuatro veces) según la sección de código a continuación porque se detectó el restablecimiento de POR.

```

int main(void)
{
    unsigned char i;
    DDRB |= (1 << PINB5);           // Set PORTB5 as output ,
    DDRB &= ~(1<<PINB7);           //Set PORTB7 as input

    if(MCUSR & 1 ){                 //POR detected
        MCUSR=0;
        for ( i=0;i<4;i++)
        {
            //LED OFF
            PORTB &= ~(1 << PINB5); // Set PORTB5 Low
            _delay_ms(300);
            //LED ON
            PORTB |= (1 << PINB5);   // Set PORTB5 On
            _delay_ms(300);
        }
    }
    else if(MCUSR & 4) {
        MCUSR=0;
    }
}

```

9

Tarea 9 - Prueba BOR

- Después de la prueba POR, deslice el interruptor del cable VCC ajustable externo hacia el lado izquierdo (2,5 V)
- Deslice el interruptor del cable VCC ajustable externo hacia el lado derecho (3,2 V)
- Resultado: el LED naranja responderá parpadeando (0,1 ms, ocho veces) según el código resaltado a continuación porque se detectó el restablecimiento de BOR.


```

else if(MCUSR & 4) { //BOR reset detected
    MCUSR=0;
    for (i=0;i<8;i++)
    {
        //LED OFF
        PORTB &= ~(1 << PINB5); // Set PORTB5 Low
        _delay_ms(100);
        //LED ON
        PORTB |= (1 << PINB5); // Set PORTB5 On
        _delay_ms(100);
    }
}

else if(MCUSR & 8) { // WDT reset detected
    MCUSR=0;
    WDT_Init();
    for (i=0;i<8;i++)
    {
        wdt_reset();
        //LED OFF
        PORTB &= ~(1 << PINB5); // Set PORTB5 Low
        _delay_ms(20);
        //LED ON
        PORTB |= (1 << PINB5); // Set PORTB5 On
        _delay_ms(80);
    }
}

```

10

Tarea 10 - Prueba WDT

- Presione el botón en el tablero durante aproximadamente 1 segundo y luego suelte el botón. Se activarán retrasos prolongados en la rutina principal, lo que provocará un tiempo de espera de WDT.

Nota: El temporizador WDT está configurado en 2 segundos.

```

if((PINB&1<<PINB7)==0){
    PORTB |= (1 << PINB5);           // Set PORTB5 high
    _delay_ms(500);
    _delay_ms(500);
    _delay_ms(500);
    _delay_ms(500);
    _delay_ms(500);
    _delay_ms(500);
    _delay_ms(500);
    _delay_ms(500);
}
wdt_reset();
}

```

- Resultado: el LED naranja responderá con un parpadeo rápido (0,02 ms encendido, 0,08 ms apagado, cuatro veces) al principio cuando el WDT interrumpe el ISR:

```

//Watchdog timeout ISR
ISR(WDT_vect)
{
    //Burst of 0.1Hz pulses
    for (uint8_t i=0;i<4;i++)
    {
        //LED OFF
        PORTB &= ~(1 << PINB5);           // Set PORTB5 Low
        _delay_ms(80);
        //LED ON
        PORTB |= (1 << PINB5);             // Set PORTB5 On
        _delay_ms(20);
    }
}

```

- Resultado: el LED naranja volverá a responder con un parpadeo rápido (0,02 ms encendido, 0,08 ms apagado, cuatro veces) a medida que se detecta el restablecimiento del WDT:

```

else if(MCUSR & 8) {                                     // WDT reset detected
    MCUSR=0;
    WDT_Init();
    for (i=0;i<8;i++)
    {
        wdt_reset();
        //LED OFF
        PORTB &= ~(1 << PINB5);                         // Set PORTB5 Low
        _delay_ms(20);
        //LED ON
        PORTB |= (1 << PINB5);                           // Set PORTB5 On
        _delay_ms(80);
    }
    MCUSR=0;
}
}

```

```

//initialize watchdog

```

Análisis

El proyecto muestra tres formas en que puede ocurrir un reinicio: POR, BOR y WDT Timeout. Cada uno tiene una aplicación única y todos pueden ejecutarse en la misma aplicación. Los ejemplos de código son solo una referencia de cómo se pueden configurar e implementar estos tipos de restablecimientos.

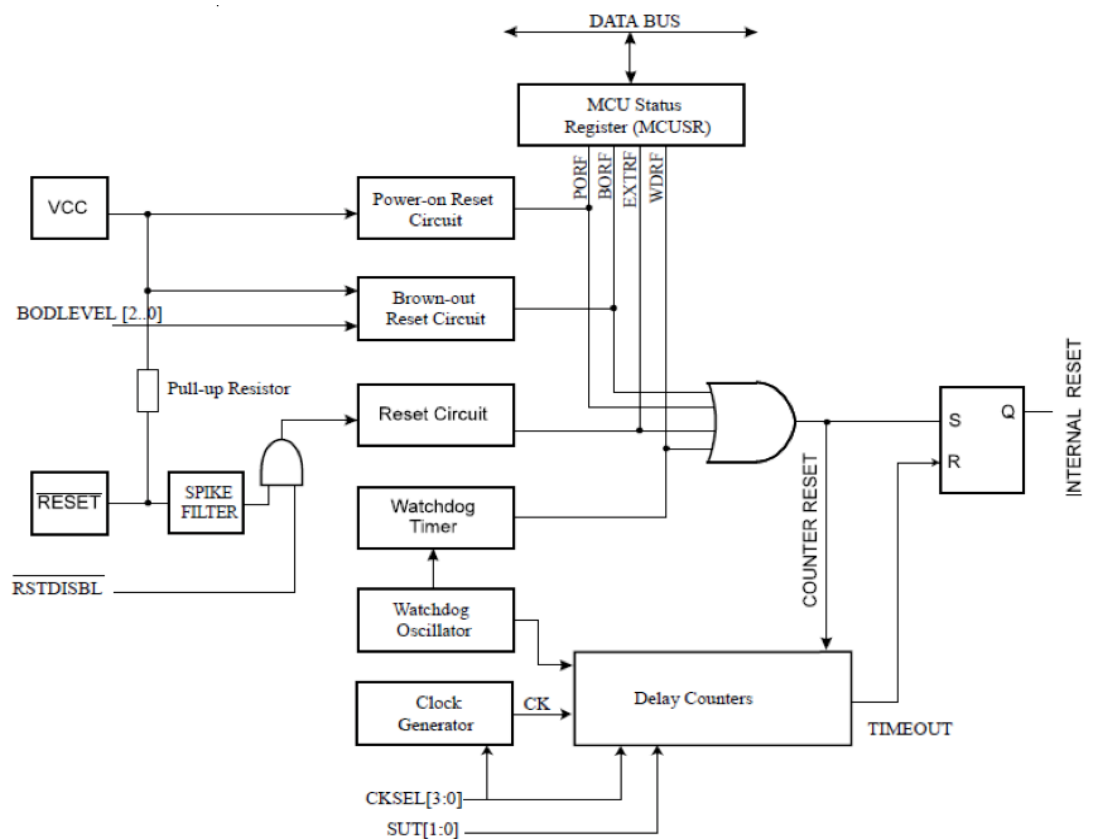
Conclusiones

El proyecto ayuda a explicar cómo funciona el circuito de reinicio dentro del AVR y cómo implementarlo. La sección de código se puede reutilizar en aplicaciones futuras que pueden requerir una estructura de reinicio similar. De ninguna manera es esta la única forma de diseñar restablecimientos en el dispositivo AVR, este es solo un proyecto de muestra simple que ayuda a explicar la operación y le permite aplicar su conocimiento y comprensión de la estructura de restablecimiento a una aplicación específica.

Fuentes de restablecimiento de AVR

El dispositivo AVR tiene cuatro fuentes de reinicio:

- **Power-on Reset** - The Microcontroller (MCU) is reset when the supply voltage is less than the Power-on Reset threshold (VPOT).
- **External Reset** - The MCU is reset when a low level is present on the RESET pin for longer than the minimum pulse length.
- **Watchdog System Reset** - The MCU is reset when the Watchdog Timer period expires and the Watchdog System Reset mode is enabled.
- **Brown-out Reset** - The MCU is reset when the supply voltage V_{CC} is less than the Brown-out Reset.



MCU Status Register (MCUSR)

To make use of the reset flags to identify a reset condition, the user should read and then reset the MCUSR as early as possible in the program. If the register is cleared before another reset occurs, the source of the reset can be found by examining the reset flags.

Name: MCUSR

Offset: 0x54

Reset: 0x00

Property: When addressing as I/O Register: address offset is 0x34

Bit	7	6	5	4	3	2	1	0
					WDRF	BORF	EXTRF	PORF
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit 3 – WDRF: Watchdog System Reset Flag

This bit is set if a Watchdog System Reset occurs. The bit is reset by a Power-on Reset, or by writing a '0' to it.

Bit 2 – BORF: Brown-out Reset Flag

This bit is set if a Brown-out Reset occurs. The bit is reset by a Power-on Reset, or by writing a '0' to it.

Bit 1 – EXTRF: External Reset Flag

This bit is set if an External Reset occurs. The bit is reset by a Power-on Reset, or by writing a '0' to it.

Bit 0 – PORF: Power-on Reset Flag

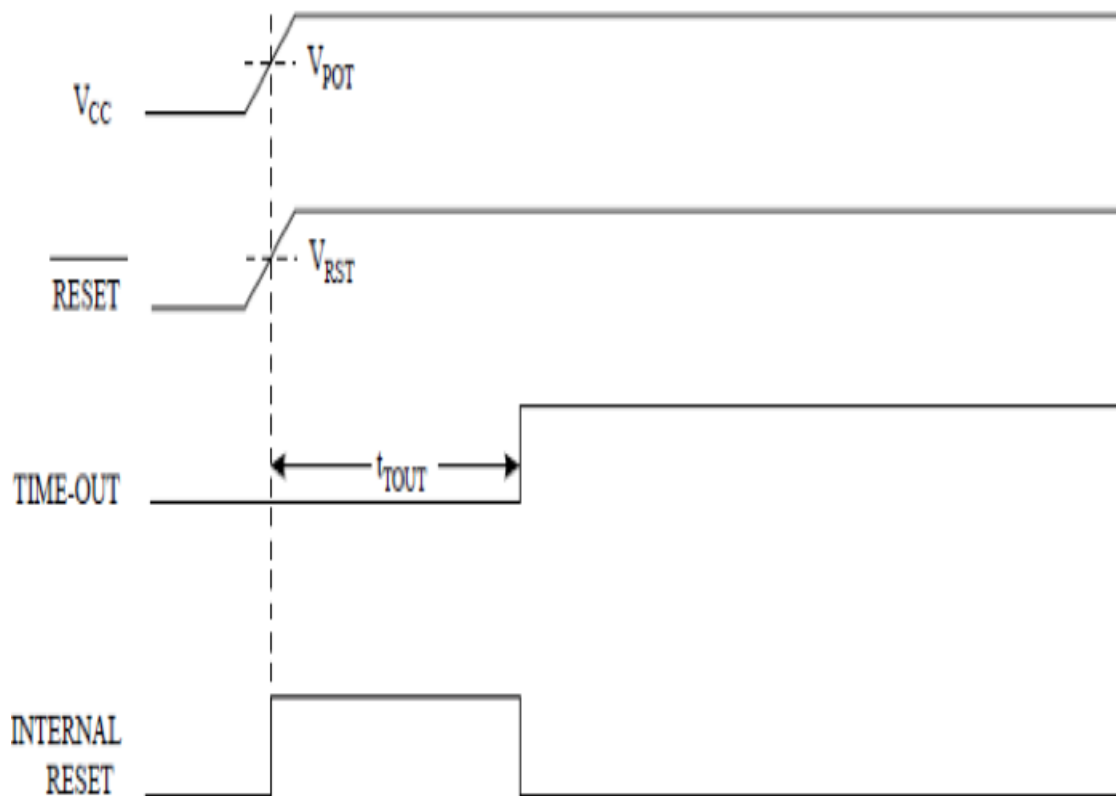
This bit is set if a Power-on Reset occurs. The bit is reset only by writing a '0' to it.

Power-on Reset (POR)

A POR pulse is generated by an On-chip detection circuit. The POR is activated whenever V_{CC} is below the detection level. The POR circuit can be used to trigger the start-up Reset, as well as to detect a failure in supply voltage.

A POR circuit ensures that the device is reset from Power-on. Reaching the Power-on Reset threshold voltage invokes the delay counter, which determines how long the device is kept in Reset after V_{CC} rise.

The Reset signal is activated again, without any delay, when V_{CC} decreases below the detection level.

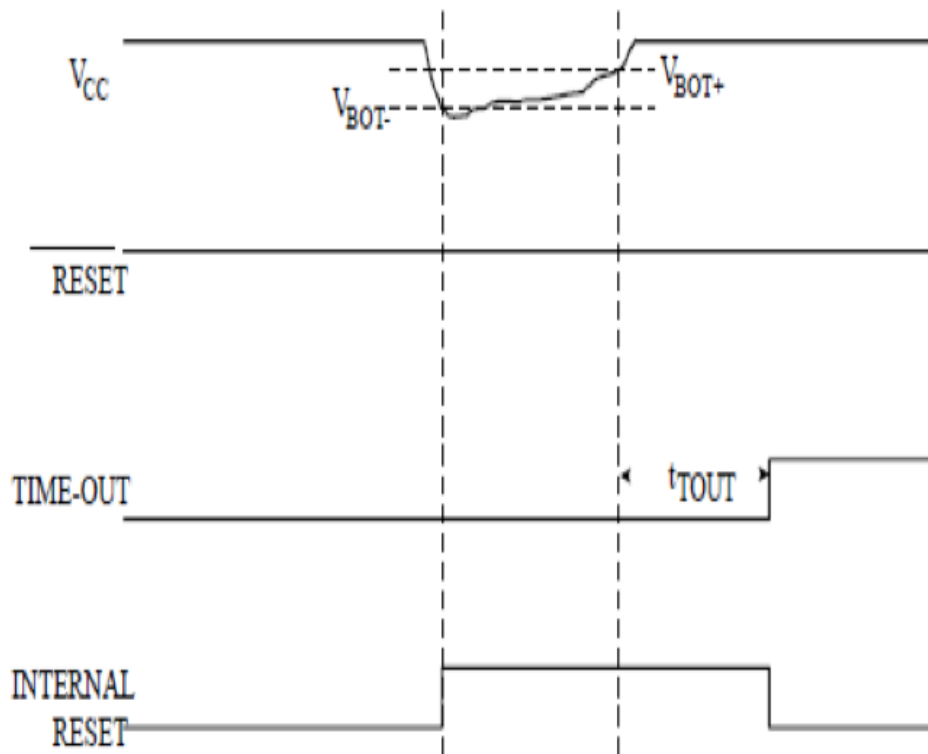


Brown-out Detection (BOD) and Brown-out Reset (BOR)

The device has an On-chip BOD circuit for monitoring the V_{CC} level during operation by comparing it to a fixed trigger level. The trigger level for the BOD can be selected by the BODLEVEL Fuses.

The BOR circuitry has hysteresis on the detection level. The BOD circuit will only detect a drop in V_{CC} if the voltage stays below the trigger level (V_{BOT-}) for longer than t_{BOD} . When that occurs, the BOR is immediately activated.

Cuando V_{CC} aumenta por encima del nivel de activación (V_{BOT+} en la siguiente figura), el contador de retardo inicia la MCU después de que haya expirado el período de tiempo de espera t_{TOUT} .



Temporizador de vigilancia (WDT)

El WDT se ejecuta independientemente del resto del sistema, lo que hace que el sistema se reinicie cada vez que se agote el tiempo de espera. Sin embargo, el software de la aplicación debe garantizar que nunca se agote el tiempo de espera reiniciando el WDT periódicamente siempre que el software se encuentre en un estado saludable conocido. Si el sistema se bloquea o la ejecución del programa se corrompe, el WDT no recibirá su reinicio periódico y, finalmente, expirará y provocará un reinicio del sistema.

El WDT mejorado en algunos dispositivos AVR también tiene la capacidad de generar interrupciones en lugar de reiniciar el dispositivo. Dado que el WDT funciona con su propio reloj independiente, se puede utilizar para activar el AVR desde todos los modos de suspensión. Esto lo convierte en un temporizador de despertador ideal, que se combina fácilmente con el funcionamiento normal como fuente de reinicio del sistema. La interrupción también se puede utilizar para obtener una advertencia temprana de un próximo restablecimiento del sistema Watchdog para que los parámetros vitales se puedan respaldar en una memoria no volátil.

Detección de fallas de reloj (CFD)

El CFD permite al usuario monitorear el oscilador de cristal de baja potencia o la señal del reloj externo (XOSC). El XOSC es monitoreado por el circuito CFD que opera con el oscilador interno de 128kHz. CFD supervisa el reloj XOSC y, si falla,

cambiará automáticamente a un reloj RC interno seguro. Cuando se produce un encendido o un restablecimiento externo, el dispositivo volverá al reloj XOSC y continuará monitoreando el reloj XOSC en busca de fallas.

El reloj seguro se deriva del reloj del sistema RC interno de 8 MHz. Esto permite configurar el reloj seguro para satisfacer las necesidades de seguridad de la aplicación.