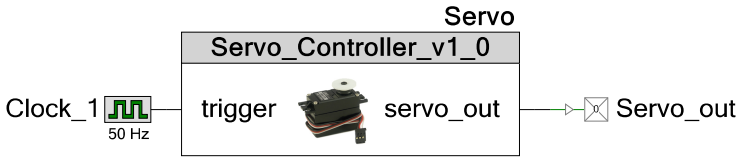
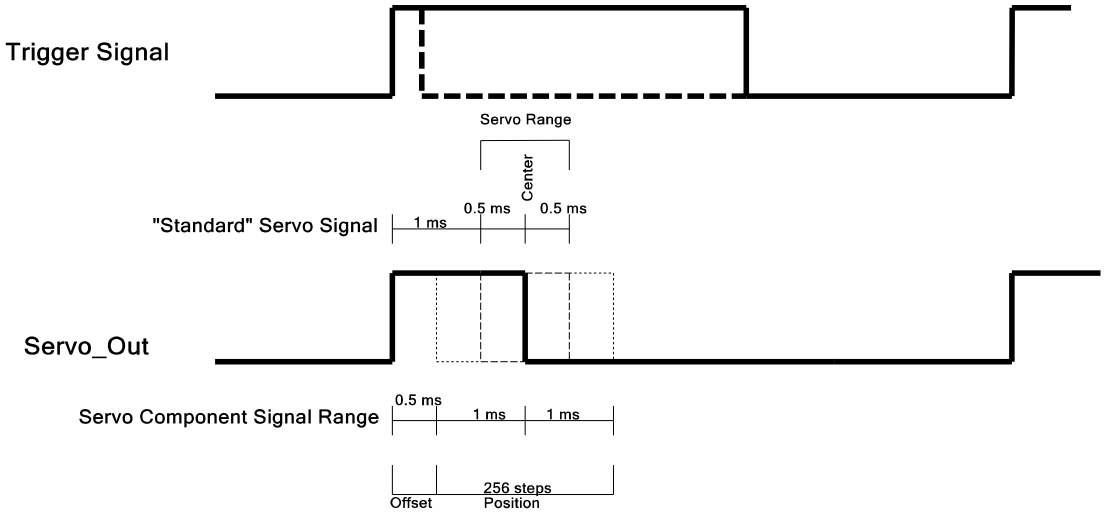


Inputs:
Trigger: initiates generating a servo pulse, Edge sensitive (rising). Place a ~ 50 hz clock on this terminal to generate a string of servo pulses sufficient for most RC servo applications. The trigger input frequency can go as high as 350 Hz (absolute maximum).

Servo_out: RC servo output pulse. with the default offset, this output can vary from 0.5 to 2.5 ms in increments of ~7.8 uS (256 steps). The offset is adjustable from 0 to 2 ms via the INSTANCE_NAME_SetOffset(uint8 offset) API.



API

INSTANCE_NAME_Start(void)
begins operation of the component. after calling Start(), the default offset and position are loaded. The next rising edge of trigger will initiate a servo pulse.

INSTANCE_NAME_Stop(void)
disables the component. All inputs on Trigger will be ignored.

INSTANCE_NAME_SetPosition(uint8 position)
sets the position of the servo by adjusting the servo pulse. A value of 127 generates a pulse 1.5 ms long (with the defualt offset loaded). a value of 0 generates a pulse 0.5 ms long (with the defualt offset) and a value of 255 generates a pulse 2.5 ms long (with the defualt offset). To calculate an exact pulse width, use the following equation:

time = 1/INSTANCE_NAME_Servo_Clock*(offset+position). *The INSTANCE_NAME_Servo_Clock is nominally 128 Khz (check the DWR resources tab for the actual frequency achieved).*

You can also write directly to the position register in the datapath by accessing INSTANCE_NAME_POSITION register. I.e. INSTANCE_NAME_POSITION=127;

INSTANCE_NAME_SetOffset(uint8 offset)

sets the initial time offset (minimum on time) for the pulse. since a standard RC servo pulse is between 1 ms and 2 ms long, the useable range is only 1 ms wide, with a minimum on time of 1 ms. The offset is adjustable so that if desired, the offset can be increased or decreased to adjust the minimum on time. the offset will be:

1/INSTANCE_NAME_Servo_Clock*offset.

You can also write directly to the Offset register in the datapath by accessing INSTANCE_NAME_OFFSET register. I.e. INSTANCE_NAME_ OFFSET=64;

uint8 position INSTANCE_NAME_GetPosition(void)
Just in case you forgot the last position you wrote to the servo, this function will remember it for you.

uint8 position INSTANCE_NAME_GetOffset(void)
Just in case you forgot the last offset you wrote to the servo, this function will remember it for you.

Oh, yeah, it uses:
1 datapath
9 product terms
4 macrocells,

Example Code (instance name 'Servo'):

```
Servo_Start();

Servo_OFFSET = 100;  // moves the offset to 0.781 ms

While(1)
{
      Servo_POSITION++;
}
```