Bus Compatible Digital PWM Controller, IXDP 610

Description

The IXDP610 Digital Pulse Width Modulator (DPWM) is a programmable CMOS LSI device which accepts digital pulse width data from a microprocessor and generates two complementary, non-overlapping, pulse width modulated signals for direct digital control of switching power bridge. The DPWM is designed to be operated under the direct control of a microprocessor and interfaces easily with most standard microprocessor and microcomputer buses. The IXDP610 is packaged in an 18-Pin slim DP.

The PWM waveform generated by the IXDP610 results from comparing the output of the Pulse Width counter to the number stored in the Pulse Width Latch (see below). A programmable "dead-time" is incorporated into the PWM waveform. The Dead-Time Logic disables both outputs on each transition of the Comparator output for the required dead-time interval.

The output stage provides complementary PWM output signals capable of sinking and sourcing 20 mA at TTL voltage levels. The Output Disable logic can be activated either by software or hardware. This facilitates cycle-by-cycle current-limit, short-circuit, over-temperature, and desaturation protection schemes.

The IXDP610 is capable of operating at PWM frequencies from zero to 390kHz; the dead-time is programmable from zero to 14 clock cycles (0 to 11 % of the PWM cycle), which allows operation with fast power MOSFETs, IGBTs, and bipolar power transistors. A trade-off between PWM frequency and resolution is provided by selecting the counter resolution to be 7-bit or 8-bit. The 20 mA output drive makes the IXDP610 capable of directly driving opto isolators and Smart Power devices. The fast response to pulse width commands is achieved by instantaneous change of the outputs to correspond to the new command. This eliminates the one-cycle delay usually associated with other digital PWM implementations.

Features

- Microcomputer bus compatible
- Two complementary outputs for direct control of a switching power bridge
- Dynamically programmable pulse width ranges from 0 to 100 %
- Two modes of operation: 7-bit or 8-bit resolution
- Switching frequency range up to 390 kHz
- Programmable Dead-time Counter prevents switching overlap
- Cycle-by-Cycle disable input to protect against over-current, over-temperature, etc.
- Outputs may be disabled under software control
- Special locking bit prevents damage to the stage in the event of a software failure
- 18-pin slim DIP package

Dimensions in inch and mm

18-Pin Slim DIP

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
<th>Maximum Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{CC}$</td>
<td>Supply voltage</td>
<td>-0.3 ... 5.5 V</td>
</tr>
<tr>
<td>$V_{IN}$</td>
<td>Input voltage</td>
<td>-0.3 ... $V_{CC}$ + 0.3 V</td>
</tr>
<tr>
<td>$V_{out}$</td>
<td>Output voltage</td>
<td>-0.3 ... $V_{CC}$ + 0.3 V</td>
</tr>
<tr>
<td>$P_{D}$</td>
<td>Maximum power dissipation</td>
<td>500 mW</td>
</tr>
<tr>
<td>$T_{stg}$</td>
<td>Storage temperature range</td>
<td>-40 ... 125 °C</td>
</tr>
</tbody>
</table>
Symbol | Definition/Condition | Characteristic Values (Over operating range, unless otherwise specified) | min. | typ. | max. \\
--- | --- | --- | --- | --- | --- \\
\(V_{CC}\) Supply voltage | | | 4.5 | 5.5 | V \\
\(T_A\) Operating free air temperature | | | -40 | 85 | °C \\
Symbol | Definition/Condition | Characteristic Values | min. | typ. | max. \\
--- | --- | --- | --- | --- | --- \\
\(V_{IH(CMOS)}\) Input High Voltage | ODIS | VCC +0.3 | | | V \\
\(V_{IL(CMOS)}\) Input Low Voltage | ODIS | | -0.3 | | V \\
\(V_H\) Input Hysteresis | ODIS | | 0.3 | 0.5 | V \\
\(V_{OH}\) Output High Voltage | OUT1 \(I_{OH} = -20\) mA | 2.4 | | | V \\
\(V_{OL}\) Output Low Voltage | OUT1 \(I_{OL} = 20\) mA | | 0.4 | | V \\
\(V_{IH(TTL)}\) Input High Voltage | All Inputs Except ODIS | 2.0 | | | VCC +0.3 V \\
\(V_{IL(TTL)}\) Input Low Voltage | All Inputs Except ODIS | | -0.3 | | 0.8 V \\
\(I_{LI}\) Input Leakage Current | | | -10 | -0.1 | 10 µA \\
\(I_{CC}\) Power Supply Current \(f_{CLK} = 5\) MHz | \(V_{IH} = V_{CC}\) or 0 | | 3.5 | | 10 mA \\
Symbol | Definition/Condition | Characteristic Values (\(T_A = 25^\circ C, V_{CC} = 5\) V ± 10%, \(C_1 = 50\) pF) | min. | typ. | max. \\
--- | --- | --- | --- | --- | --- \\
No. see Fig. 3-6 | SEL Stable to WR Low | 5 | 10 | ns \\
2 | SEL Stable after WR High | 10 | 20 | ns \\
3 | CS Low to WR Low | 5 | 10 | ns \\
4 | CS High after WR High | 5 | 10 | ns \\
5 | WR Pulse Width | 8 | 20 | ns \\
6 | Data Valid to WR High | 5 | 10 | ns \\
7 | Data Held after WR High | 10 | 20 | ns \\
8 | Clock Frequency | 50’ | 0 | 50’ MHz \\
9 | Clock Pulse Duration Low | 12.5 | 12.5 | ns \\
10 | Clock Pulse Duration High | 12.5 | 12.5 | ns \\
11 | CLK to Output when Writing to PW latch | \(5 + \frac{1}{2} T_{CLK}^{**}\) | 5 | 10 | ns \\
12 | ODIS Low to Output Low | 20 | 50 | ns \\
13 | WR High to Output Low When Writing Stop to the Control latch | 30 | 60 | ns \\
14 | RST Low Time | 50 | ns \\

* Output will change 1 rising CLOCK edge +5ns after WR (see Fig. 6) 
** \(T_{CLK} = \frac{1}{f_{CLK}}\)
**Nomenclature of Digital PWM Controller**

### Pin Description IXDP 610PI

<table>
<thead>
<tr>
<th>Sym.</th>
<th>Pin Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D0</td>
<td>DATA BUS</td>
<td>the data bus on the IXDP610 is configured for input only. Data to be written to the IXDP610 is placed on data lines D0 through D7 during a microprocessor write cycle. Data is accepted by the IXDP610 when CHIP SELECT is low and the WRITE input goes from a low to a high state. The SELECT input determines whether the data written to the IXDP610 goes to the internal Pulse Width (PW) latch or to the Control latch. D0 is the least significant bit and D7 is the most significant bit.</td>
</tr>
<tr>
<td>D1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GND</td>
<td>CIRCUIT GROUND</td>
<td></td>
</tr>
<tr>
<td>OUT2</td>
<td>COMPLEMENTARY OUTPUTS</td>
<td>these two outputs provide the complementary PWM signals. The base period or cycle time of these outputs is determined by the CLOCK and the control latch.</td>
</tr>
<tr>
<td>OUT1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vcc</td>
<td>POWER SUPPLY (5 V ± 10 %)</td>
<td></td>
</tr>
<tr>
<td>CLK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ODIS</td>
<td>OUTPUT DISABLE</td>
<td>asserting this Schmitt trigger input forces the complementary outputs to be immediately disabled (OUT1 and OUT2 are forced low). The complementary outputs will remain low as long as this input is asserted, and for the duration of the PWM cycle in which OUTPUT DISABLE goes from low to high; i.e., the complementary outputs are not re-enabled until the beginning of the next PWM cycle, and then only if OUTPUT DISABLE and the Stop bit in the Control latch are not asserted.</td>
</tr>
</tbody>
</table>

### IXDP 610 PI (Example)

**IX** — IXYS

**DP 610** — Digital PWM Controller

**P** — 18-Pin Plastic DIP

**I** — Industrial

(-40 to 85°C)
Description

Introduction

The IXDP610 is a digital PWM controller. It simplifies the interface between a microprocessor and a switching power bridge by providing to a micro-processor the means to directly control the average voltage across a load (DC motor, etc.). Since the IXDP610 generates two complementary PWM control signals, there is no need for Digital to Analog Converters (DACs), Sawtooth Generators, and Analog Comparators. OUT1 and OUT2 can directly drive the buffers to the power transistors.

Use of the IXDP610 in a DC servo system is depicted in the system block diagram shown in Fig. 1. The IXDP610 receives digital data from the microprocessor and converts the data to a pair of complementary PWM signals that can be used to control the average voltage across a DC servo motor. A Shaft Encoder Peripheral Interface (SEPI) IC converts incremental encoder signals to a binary number so the micro-processor can monitor and complete the control of the DC servo motor.

It is possible to generate PWM control signals in software with a dedicated microprocessor or microcontroller. This has the limitation, however, of very low switching frequencies (<5 kHz) and significant software overhead. By using the IXDP610 to handle the generation of the PWM control signals, a microprocessor can handle several PWM channels and the PWM control signals can switch at relatively high rates (up to 300 kHz). Servicing the IXDP610 is as simple as writing an 8-bit number to the Pulse Width latch whenever a change in duty cycle is desired. This is analogous to writing data to a DAC.

Programmable dead-time

Because the IXDP610 is a digital IC, and is programmable, it is possible to tailor the dead-time period (defined as \( t_{\text{dt}} \) in Fig. 2). IXDP610’s programmable dead-time feature is difficult to duplicate in the equivalent analog system. The control of a switching bridge usually involves a process of alternating the “on-time” of two power switches connected in series between a high-voltage and a low-voltage. For example, the H-bridge of Fig. 3 can be operated by turning the upper left and lower right transistors on and leaving the two remaining transistors off, during the first half of the PWM cycle. In the second half of the cycle, the upper right and lower left transistors are on and the remaining two are off. During the transition, between the first half and the second half of the PWM cycle, there is a very short period of time when both the upper transistor and the lower transistor in a leg could be on. If both transistors are on, for this short period of time, they will effectively short the high voltage supply to ground—this is an undesirable situation.

The IXDP610’s programmable dead-time feature prevents this situation by guaranteeing that both transistors in a leg are off for a minimum of time during a transition (the dead-time period). Since the dead-time is programmable, it can be tailored to the specific application. It can be short for high-speed MOSFETs and longer for IGBTs.

Protection circuitry

The IXDP610 has several features that facilitate protection of the power devices being controlled. The ODIS pin is an input that can be driven by external hardware under emergency shutdown conditions, such as over-current and over-temperature. The Stop bit, in the Control latch, provides a mechanism through which the software can indefinitely disable the complementary outputs. ODIS and Stop perform similar functions, they provide a means to protect the power devices from measurable system hazards such as over-current, over-voltage, over-temperature etc.

Software runaway is a system hazard that is difficult or impossible to measure. The Lock bit, in the Control latch, can be used to protect the system from software runaway and/or errors. Setting the Lock bit prevents subsequent writes to the Control latch from having any affect on the IXDP610’s operating parameters.

Control latch

The Control latch is composed of eight bits that determine the IXDP610’s
Dead-time counter bits - these three bits determine the dead-time period, as defined by Fig. 2. Dead-time is that period of time when both OUT1 and OUT2 are low. Any binary number from 000 through 111 is valid. Thus, eight different dead-time periods can be programmed. DT0 is the least significant bit and DT2 is the most significant bit. A 000 binary means no dead-time and a 111 means maximum dead-time. Each dead-time count corresponds to two CLOCK periods. For instance, if a binary three (3) is programmed into the dead-time bits, the dead-time will be six external CLOCK cycles long.

The dead-time is provided to aid in preventing switch overlap. The Dead-time Counter delays turning on the switch connected to OUT1 until the switch connected to OUT2 has had sufficient time to turn off. The complement is also true, the dead-time counter delays turning on the switch connected to OUT2 until the switch connected to OUT1 has had sufficient time to turn off. Since the dead-time counter is programmable, the user can optimize the dead-time delay to suit their specific application.

In a typical PWM cycle (refer to Fig. 2) two dead-time periods will occur. One follows the turnoff of OUT2. The dead-time counter is triggered by an output turning off. During a dead-time period, both outputs are guaranteed to be off (no dead-time periods occur during 0% and 100% duty-cycle states). The dead-time period overlaps the ontime of an output, therefore, it shortens the on-time without affecting the base PWM cycle time. A dead-time period is only inserted if an output changes from high to low (on to off). Thus, if a PWM duty cycle is chosen such that an output would be on for a period of time equal to or less than one dead-time period, the switch associated with that output will not be turned on during the PWM cycle. In this special case, one will observe only dead-time period per PWM cycle time, rather than the two dead-time periods shown in Fig. 2.

<table>
<thead>
<tr>
<th>SEL</th>
<th>CS</th>
<th>WR</th>
<th>Resulting Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>1</td>
<td>0</td>
<td>No Action</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>Load D0-D7 into PW latch</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>Load D0-D7 into Control Latch</td>
</tr>
</tbody>
</table>

Table 1 Bus Transaction Truth Table

operating parameters. Those bits are summarized in Table 2.

The following formula can be used to determine the PWM base period:

If \(((\frac{7}{8} \text{ bit} = 0) \text{ And } (\text{DIV bit} = 0))\), the PWM base period \(= \text{CLOCK period} \times 128\) else if \(((\frac{7}{8} \text{ bit} = 0) \text{ And } (\text{DIV bit} = 1))\), the PWM base period \(= \text{CLOCK period} \times 256\) else if \(((\frac{7}{8} \text{ bit} = 1) \text{ And } (\text{DIV bit} = 0))\), the PWM base period \(= \text{CLOCK period} \times 256\) else if \(((\frac{7}{8} \text{ bit} = 1) \text{ And } (\text{DIV bit} = 1))\), the PWM base period \(= \text{CLOCK period} \times 512\)

The Pulse Width number that is written to the Pulse Width latch represents the high time of OUT1 (the low time of OUT2). The Dead-time Counter decreases the on-time (output high) of an output by one dead-time period \(t_{\text{DT}}\).

Stop bit - writing a zero to this bit immediately disables the complementary outputs (OUT1 and OUT2 are forced to zero). As long as this bit is a zero, the complementary outputs will be disabled. This bit is not affected by the Lock bit. This bit is equivalent in function to the OUTPUT DISABLE input. The outputs will not be re-enabled until the start of the PWM period which has both the Stop bit and the OUTpuT DISABLE input set to ones.

PW latch - The binary number written to the PW latch represents the duty cycle of the complementary PWM outputs. Percent duty cycle is defined as follows:

\[
\text{Percent duty cycle} = \left(\frac{\text{time at 1}}{\text{time at 0}}\right) \times 100
\]

Table 2 Control Latch Bits
The Resolution bit in the Control latch determines whether the number in the PW latch has 7 significant bits or 8 significant bits. The following formulae can be used to determine the resulting PWM waveform's duty cycle:

For 7-bit mode operation:
\[
\text{% duty cycle} = \frac{\text{PW number}}{128} \times 100
\]

For 8-bit mode operation:
\[
\text{% duty cycle} = \frac{\text{PW number}}{256} \times 100
\]

The formulae are valid for all PW numbers except those at the extremes. The following table illustrates the resulting percent duty cycle for several PW numbers. (The complete table would have 256 entries, those entries that have been omitted can be calculated using the above formulae.)

The PWM duty cycle byte can be written to at any time. If the outputs are disabled by either the Stop bit in the Control latch or the OUTPUT DISABLE input, writing to the PWM duty cycle byte will have no effect on the outputs. When the outputs are re-enabled, the duty cycle will be determined by the last byte written to the PWM duty cycle byte.

### Application Information

#### Introduction

The IXDP610 is a digital PWM controller intended for use with general-purpose microprocessors and microcontrollers. Therefore, it is important to understand how the microprocessor hardware and software interacts with and affects the operation of the IXDP610. On the following pages one will find discussions of some of the most important hardware and software interface issues. Among these issues are the hardware interface, how to choose the IXDP610's clock, initialization of the DPWM, the effect of the dead-time on the duty cycle, and the response of the IXDP610 to changes in the Pulse Width latch number. The following pages should be studied carefully by both the hardware and the software designer.

The IXDP610 can be interfaced with virtually any microprocessor or microcontroller. Some interface examples are shown below.

#### Frequency and dead-time considerations

Typical applications for the IXDP610 include full and half bridge systems. Shown in Fig. 3 is a full bridge system. The programmable dead-time feature of the IXDP610 aids in preventing shorts in the power bridge and allows use of either fast MOSFETs or slower IGBTs and bipolar transistors. Table 4 shows some of the PWM frequency and dead-time combinations that can be obtained with the IXDP610. The various options shown in the table are selected by varying the CLK frequency and the Divide and 7/8 bit in the

---

**8051 to IXDP610 Interface**

Fig. 4 is an example of how the IXDP610 can be interfaced with an Intel 8051 microcontroller. The interface is very simple and is ideally suited for servo motor control applications. The 11.059 MHz clock allows one to use the 8051's built-in serial communication hardware at any standard baud rate. At this clock frequency, the IXDP610 can be configured for a 21.6 kHz switching frequency and a dead-time between zero and 1.26 \(\mu\)s, which is adjustable in 180 ns steps.

**8088 to IXDP610 Interface**

Fig. 5 is just one example of how the IXDP610 can be interfaced with the Intel 8088 microprocessor. Using a 5 MHz clock (15 MHz crystal) the IXDP610 can be configured for a 19.53 kHz switching frequency. The deadtime can be adjusted between 0 and 2.8 \(\mu\)s, in 400 ns steps. This configuration is ideally suited for driving DC servo motor amplifiers that use MOSFET, IGBT, or bipolar transistors.
IXDP610’s Control latch. The “%” columns express the dead-time as a percent of the PWM cycle time.

If a zero is written to the 7/8 bit, the IXDP610 is programmed for 7-bit resolution, writing a one programs the IXDP610 for 8-bit resolution. If a one is written to the Divide bit, the external clock (CLK) is divided by two before being presented to the Pulse Width counter; a zero in the Divide bit passes CLK directly to the Pulse Width Counter with no division of the frequency. For a given CLK frequency one can select three different PWM frequencies: CLK/128, CLK/256, and CLK/512. (CLK/256 can be selected for either 7-bit or 8-bit resolution.

Software Considerations

Initialization and the Lock Bit

After power-up, the IXDP610 should be reset via the RST input. Doing so will guarantee the initial state of the DPWM and effectively write a 01000111 binary to the Control latch. Thus, after asserting RST, the IXDP610 is set to the following state:

- **Stop** is asserted, disabling OUT1 and OUT2
- 8-bit resolution is selected
- CLK is divided by one (not divided by two)
- Lock bit is “UNLOCKED”
- Dead-time Counter is set for maximum dead-time.

Asserting RST is the only means by which the Lock bit can be “unlocked”. The lock bit must be cleared in order to write to all other bits in the Control latch, except the Stop bit.

The IXDP610 does not undergo an internal reset on power-up; therefore, it is recommended that the system reset be connected to the DPWM, as in Fig. 5. If one wishes to allow software control over the RST input, they should “OR” the system reset and an I/O bit together, so the DPWM has a known state following system reset. Before initializing the Control latch, one should first write a valid number to the Pulse Width latch (i.e., a number that results in 0 V applied to the load). Asserting RST clears the Pulse Width latch.

During a write to the Control latch, all bits can be modified simultaneously, including the Lock bit. Thus, only one write is necessary to set the dead-time: 1) assert the Lock bit; 2) choose the Divide bit state; 3) choose the resolution. In most applications it is not necessary to change the dead-time, the Divide bit, or the 7/8 bit “on the fly”. Therefore, it is recommended that the Lock bit be asserted during initialization of the Control latch. Setting the Lock bit guarantees that a software runaway will not modify the state of the dead-time bit, thereby preventing an accidental short of the bridge. If the RST input is accessible to the software (via an I/O bit, spare chip select, etc.), the hardware associated with asserting the RST input should be designed to minimize the possibility of resetting the IXDP610 in the event of a software runaway, since asserting the RST input clears the Lock bit, allowing modification of the DPWM’s Control latch.

**Software Overflow Protection**

In many applications, the Pulse Width number written by the microprocessor to the IXDP610’s Pulse Width latch is the result of closed-loop numeric calculations. Depending on the algorithm used, the calculated PWM number may be susceptible to overflow, i.e. the calculated PWM number could be larger than the available 8-bits (or 7-bits) provided in the Pulse Width latch. If this is the case, it is important that the software checks for overflow conditions before writing a number to the Pulse Width latch. Following is an example assuming 8-bit resolution:

if (PWM_num < 0), check for underflow, PWM_num = 0, set to minimum limit
else if (PWM_num > 255), check for overflow, PWM_num = 255; set to maximum limit

**Effect of Dead-time on Duty Cycle**

The IXDP610 has been designed to generate PWM signals that range from 0 % to 100 %, inclusive. When zero dead-time has been selected (by writing 000 to the dead-time bits) the duty cycle of a PWM cycle can be determined by using the formulae shown on page 32/33. Fig. 6 illustrates the effect that a nonzero dead-time has on the PWM waveform.

The dead-time feature built into the IXDP610 guarantees that both OUT1 and OUT2 remain off for the duration of the dead-time period. A dead-time period occurs each time either OUT1 or OUT2 turns off; the dead-time period overlaps the on-time of an output (see Fig. 6c). Thus, if the desired duty cycle is such that the

<table>
<thead>
<tr>
<th>PWM Frequency kHz</th>
<th>Min. %</th>
<th>Step %</th>
<th>Max. %</th>
<th>CLK MHz</th>
<th>7/8 bit</th>
<th>DIV bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>0</td>
<td>1.56</td>
<td>0.052</td>
<td>10.9</td>
<td>0.363</td>
<td>38.4</td>
</tr>
<tr>
<td>200</td>
<td>0</td>
<td>1.56</td>
<td>0.078</td>
<td>10.9</td>
<td>0.547</td>
<td>25.6</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>0.78</td>
<td>0.078</td>
<td>5.5</td>
<td>0.547</td>
<td>25.6</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>1.56</td>
<td>0.156</td>
<td>10.9</td>
<td>1.094</td>
<td>12.8</td>
</tr>
<tr>
<td>50</td>
<td>0</td>
<td>0.39</td>
<td>0.078</td>
<td>2.7</td>
<td>0.547</td>
<td>25.6</td>
</tr>
<tr>
<td>50</td>
<td>0</td>
<td>0.78</td>
<td>0.156</td>
<td>5.5</td>
<td>1.094</td>
<td>12.8</td>
</tr>
<tr>
<td>50</td>
<td>0</td>
<td>0.78</td>
<td>0.156</td>
<td>5.5</td>
<td>1.094</td>
<td>12.8</td>
</tr>
<tr>
<td>50</td>
<td>0</td>
<td>1.56</td>
<td>0.312</td>
<td>10.9</td>
<td>2.188</td>
<td>6.4</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>0.39</td>
<td>0.195</td>
<td>2.7</td>
<td>1.367</td>
<td>10.24</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>0.78</td>
<td>0.391</td>
<td>5.5</td>
<td>2.734</td>
<td>5.12</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>0.78</td>
<td>0.391</td>
<td>5.5</td>
<td>2.734</td>
<td>5.12</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>1.56</td>
<td>0.781</td>
<td>10.9</td>
<td>5.469</td>
<td>2.56</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0.39</td>
<td>0.781</td>
<td>2.7</td>
<td>5.469</td>
<td>2.56</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0.78</td>
<td>1.562</td>
<td>5.5</td>
<td>10.94</td>
<td>1.28</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0.78</td>
<td>1.562</td>
<td>5.5</td>
<td>10.94</td>
<td>1.28</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1.56</td>
<td>3.125</td>
<td>10.9</td>
<td>21.88</td>
<td>0.64</td>
</tr>
</tbody>
</table>

Table 4. Sample PWM Frequency and Dead-time Options
have only one dead-time period inserted in each PWM cycle. In Fig. 6b the desired ontime of OUT1 is less than the one dead-time period, therefore OUT1 can never turn on. The same is true for OUT2 in Fig. 6d. Fig. 6c is the normal situation, where both outputs turn on and off during one PWM cycle and, as a result, two dead-time periods are inserted.

### Response to a Change in the Pulse Width Number

One can change the Pulse Width number at any time. It is not necessary to synchronize writes to the Pulse Width latch with the CLK or the PWM cycle period. The IXDP610 responds to the new Pulse Width number three clock cycles after the Pulse Width latch is loaded (1 CLK cycle after WR goes high). Thus, OUT1 and OUT2 will immediately reflect the new Pulse Width number. The IXDP610 does not wait until the next PWM cycle to implement a change in the Pulse Width number. (See Fig. 7).

The resulting duty cycle is somewhere between the old and the new duty cycle. The exact value of the resulting duty cycle depends on when the Width Latch is loaded (1 CLK cycle after WR goes high). Thus, OUT1 and OUT2 will immediately reflect the new Pulse Width number. The IXDP610 does not wait until the next PWM cycle to implement a change in the Pulse Width number.

Fig. 7a shows what happens when the Pulse Width number is changed from 20 % to 80 % near the middle of the PWM cycle. Fig. 7b shows the reverse situation.

The resulting duty cycle is somewhere between the old and the new duty cycle. The exact value of the resulting duty cycle depends when the Width latch is loaded (1 CLK cycle after WR goes high). Thus, OUT1 and OUT2 will immediately reflect the new Pulse Width number. The IXDP610 does not wait until the next PWM cycle to implement a change in the Pulse Width number.