11.7. List of Registers

The PIO0 and PIO1 registers start at base addresses of 0x50200000 and 0x50300000 respectively (defined as PIO0_BASE and PIO1_BASE in SDK).

Table 980. List of PIO registers

0x000	CTRL	PIO control register
0x004	FSTAT	FIFO status register
0x008	FDEBUG	FIFO debug register
0x00c	FLEVEL	FIFO levels
0x010	TXF0	Direct write access to the TX FIFO for this state machine. Each write pushes one word to the FIFO. Attempting to write to a full FIFO has no effect on the FIFO state or contents, and sets the sticky FDEBUG_TXOVER error flag for this FIFO. このステートマシンのTX FIFOへの直接書き込みアクセス。各書き込みは、FIFOに1ワードをプッシュする。満杯のFIFOに書き込みを試みても、FIFOの状態や内容には何の影響もなく、このFIFOに対してスティッキーなFDEBUG_TXOVERエラーフラグが設定される。
0x014	TXF1	Direct write access to the TX FIFO for this state machine. Each write pushes one word to the FIFO. Attempting to write to a full FIFO has no effect on the FIFO state or contents, and sets the sticky FDEBUG_TXOVER error flag for this FIFO. (同上)
0x018	TXF2	Direct write access to the TX FIFO for this state machine. Each write pushes one word to the FIFO. Attempting to write to a full FIFO has no effect on the FIFO state or contents, and sets the sticky FDEBUG_TXOVER error flag for this FIFO. (同上)
0x01c	TXF3	Direct write access to the TX FIFO for this state machine. Each write pushes one word to the FIFO. Attempting to write to a full FIFO has no effect on the FIFO state or contents, and sets the sticky FDEBUG_TXOVER error flag for this FIFO. (同上)
0x020	RXF0	Direct read access to the RX FIFO for this state machine. Each read pops one word from the FIFO. Attempting to read from an empty FIFO has no effect on the FIFO state, and sets the sticky FDEBUG_RXUNDER error flag for this FIFO. The data returned to the system on a read from an empty FIFO is undefined. このステート マシンの RX FIFO への直接読み出しアクセス。各読み取りは、FIFOから1ワードをポップします。空のFIFOから読み出そうとしてもFIFOの状態には影響せず、このFIFOのスティッキーなFDEBUG_RXUNDERエラーフラグが設定されます。空のFIFOからの

		読み取りでシステムに返されるデータは未定義である。
0x024	RXF1	Direct read access to the RX FIFO for this state machine. Each read pops one word from the FIFO. Attempting to read from an empty FIFO has no effect on the FIFO state, and sets the sticky FDEBUG_RXUNDER error flag for this FIFO. The data returned to the system on a read from an empty FIFO is undefined. (同上)
0x028	RXF2	Direct read access to the RX FIFO for this state machine. Each read pops one word from the FIFO. Attempting to read from an empty FIFO has no effect on the FIFO state, and sets the sticky FDEBUG_RXUNDER error flag for this FIFO. The data returned to the system on a read from an empty FIFO is undefined. (同上)
0x02c	RXF3	Direct read access to the RX FIFO for this state machine. Each read pops one word from the FIFO. Attempting to read from an empty FIFO has no effect on the FIFO state, and sets the sticky FDEBUG_RXUNDER error flag for this FIFO. The data returned to the system on a read from an empty FIFO is undefined. (同上)
0x030	IRQ	State machine IRQ flags register. Write 1 to clear. There are eight state machine IRQ flags, which can be set, cleared, and waited on by the state machines. There's no fixed association between flags and state machines --- any state machine can use any flag. Any of the eight flags can be used for timing synchronisation between state machines, using IRQ and WAIT instructions. Any combination of the eight flags can also routed out to either of the two system-level interrupt requests, alongside FIFO status interrupts --- see e.g. IRQ0_INTE. ステート・マシン IRQ フラグ・レジスタ。クリアするには1を書き込む。ステート・マシンIRQフラグは8つあり、ステート・マシンによってセット、クリア、待機させることができる。フラグとステート・マシンの間に固定的な関連はない。--- どんなステートマシンでも、どんなフラグでも使うことができる。 IRQ命令とWAIT命令を使用して、8つのフラグのいずれかをステートマシン間のタイミング同期に使用できます。8つのフラグの組み合わせは、FIFOステータス割り込みと並んで、2つのシステムレベル割り込み要求のいずれかにルーティングすることもできます。例えば、IRQ0_INTE を参照のこと。
0x034	IRQ_FORCE	Writing a 1 to each of these bits will forcibly assert the corresponding IRQ. Note this is different to the INTF register: writing here affects PIO internal state. INTF just asserts the processor-facing IRQ signal for testing ISRs, and is not visible to the state machines.

		これらの各ビットに1を書き込むと、対応するIRQが強制的にアサートされる。これはINTFレジスタとは異なることに注意。ここに書き込むとPIOの内部状態に影響する。INTFは、ISRをテストするためのプロセッサ向けIRQ信号をアサートするだけで、ステート・マシンからは見えない。
0x038	INPUT_SYNC_BYPASS	There is a 2-flipflop synchronizer on each GPIO input, which protects PIO logic from metastabilities. This increases input delay, and for fast synchronous IO (e.g. SPI) these synchronizers may need to be bypassed. Each bit in this register corresponds to one GPIO. 0 → input is synchronized (default) 1 → synchronizer is bypassed If in doubt, leave this register as all zeroes. 各GPIO入力には2フリップフロップのシンクロナイザーがあり、PIOロジックをメタスタビリティから保護します。これは入力遅延を増加させ、高速同期IO（例えばSPI）ではこれらのシンクロナイザーをバイパスする必要があるかもしれません。このレジスタの各ビットは1つのGPIOに対応する。 0 → 入力は同期される（デフォルト） 1 → 同期器はバイパスされる 疑わしい場合は、このレジスタをすべてゼロのままにしてください。
0x03c	DBG_PADOUT	Read to sample the pad output values PIO is currently driving to the GPIOs. On RP2040 there are 30 GPIOs, so the two most significant bits are hardwired to 0. PIOが現在GPIOにドライブしているパッド出力値をサンプリングするために読み出します。RP2040には30個のGPIOがあるので、最上位2ビットは0にハードワイヤされています。
0x040	DBG_PADOE	Read to sample the pad output enables (direction) PIO is currently driving to the GPIOs. On RP2040 there are 30 GPIOs, so the two most significant bits are hardwired to 0. PIOが現在GPIOを駆動しているパッド出力イネーブル（方向）をサンプリングするために読み出します。RP2040には30個のGPIOがあるので、最上位2ビットは0にハードワイヤされています。
0x044	DBG_CFGINFO	The PIO hardware has some free parameters that may vary between chip products. These should be provided in the chip datasheet, but are also exposed here. PIOハードウェアには、チップ製品によって異なる自由パラメータがあります。これらはチップのデータシートに記載されているはずですが、ここでも公開しています。
0x048	INSTR_MEM0	Write-only access to instruction memory location 0 命令メモリ・ロケーション0への書き込み専用アクセス
0x04c	INSTR_MEM1	命令メモリ・ロケーション1への書き込み専用アクセス

0x050	INSTR_MEM2	命令メモリ・ロケーション2への書き込み専用アクセス
0x054	INSTR_MEM3	命令メモリ・ロケーション3への書き込み専用アクセス
0x058	INSTR_MEM4	命令メモリ・ロケーション4への書き込み専用アクセス
0x05c	INSTR_MEM5	命令メモリ・ロケーション5への書き込み専用アクセス
0x060	INSTR_MEM6	命令メモリ・ロケーション6への書き込み専用アクセス
0x064	INSTR_MEM7	命令メモリ・ロケーション7への書き込み専用アクセス
0x068	INSTR_MEM8	命令メモリ・ロケーション8への書き込み専用アクセス
0x06c	INSTR_MEM9	命令メモリ・ロケーション9への書き込み専用アクセス
0x070	INSTR_MEM10	命令メモリ・ロケーション10への書き込み専用アクセス
0x074	INSTR_MEM11	命令メモリ・ロケーション11への書き込み専用アクセス
0x078	INSTR_MEM12	命令メモリ・ロケーション12への書き込み専用アクセス
0x07c	INSTR_MEM13	命令メモリ・ロケーション13への書き込み専用アクセス
0x080	INSTR_MEM14	命令メモリ・ロケーション14への書き込み専用アクセス
0x084	INSTR_MEM15	命令メモリ・ロケーション15への書き込み専用アクセス
0x088	INSTR_MEM16	命令メモリ・ロケーション16への書き込み専用アクセス
0x08c	INSTR_MEM17	命令メモリ・ロケーション17への書き込み専用アクセス
0x090	INSTR_MEM18	命令メモリのロケーション18への書き込み専用アクセス
0x094	INSTR_MEM19	命令メモリのロケーション19への書き込み専用アクセス
0x098	INSTR_MEM20	命令メモリのロケーション20への書き込み専用アクセス
0x09c	INSTR_MEM21	命令メモリのロケーション21への書き込み専用アクセス
0x0a0	INSTR_MEM22	命令メモリのロケーション22への書き込み専用アクセス
0x0a4	INSTR_MEM23	命令メモリのロケーション23への書き込み専用アクセス
0x0a8	INSTR_MEM24	命令メモリのロケーション24への書き込み専用アクセス
0x0ac	INSTR_MEM25	命令メモリのロケーション25への書き込み専用アクセス
0x0b0	INSTR_MEM26	命令メモリのロケーション26への書き込み専用アクセス
0x0b4	INSTR_MEM27	命令メモリのロケーション27への書き込み専用アクセス
0x0b8	INSTR_MEM28	命令メモリのロケーション28への書き込み専用アクセス
0x0bc	INSTR_MEM29	命令メモリのロケーション29への書き込み専用アクセス
0x0c0	INSTR_MEM30	命令メモリのロケーション30への書き込み専用アクセス
0x0c4	INSTR_MEM31	命令メモリのロケーション31への書き込み専用アクセス
0x0c8	SM0_CLKDIV	ステートマシン用分周器レジスタ 0 Frequency = clock freq / (CLKDIV_INT + CLKDIV_FRAC / 256)
0x0cc	SM0_EXECCTRL	Execution/behavioural settings for state machine 0 ステートマシン0の実行/動作設定
0x0d0	SM0_SHIFTCTRL	Control behaviour of the input/output shift registers for state machine 0 ステートマシン0の入出力シフトレジスタの制御動作
0x0d4	SM0_ADDR	Current instruction address of state machine 0 ステートマシン0の現在の命令アドレス
0x0d8	SM0_INSTR	Read to see the instruction currently addressed by state machine 0's program counter Write to execute an instruction immediately (including jumps) and then resume execution. ステート・マシン0のプログラム・カウンタが現在アドレス指定している命令を確認するための読み出し。 命令を即座に実行し（ジャンプを含

		む)、その後実行を再開するための書き込み。
0x0dc	SM0_PINCTRL	State machine pin control ステートマシン0のピン制御
0x0e0	SM1_CLKDIV	ステートマシン1用分周レジスタ。 Frequency = clock freq / (CLKDIV_INT + CLKDIV_FRAC / 256)
0x0e4	SM1_EXECCTRL	ステートマシン1の入出力シフトレジスタの実行/動作設定
0x0e8	SM1_SHIFTCTRL	ステートマシン1の入出力シフトレジスタの制御動作
0x0ec	SM1_ADDR	ステートマシン1の現在の命令アドレス
0x0f0	SM1_INSTR	ステート・マシン1のプログラム・カウンタが現在アドレス指定している命令を確認するための読み出し。 命令を即座に実行し（ジャンプを含む)、その後実行を再開するための書き込み。
0x0f4	SM1_PINCTRL	ステートマシン1のピン制御
0x0f8	SM2_CLKDIV	ステートマシン2用分周レジスタ。 Frequency = clock freq / (CLKDIV_INT + CLKDIV_FRAC / 256)
0x0fc	SM2_EXECCTRL	ステートマシン2の入出力シフトレジスタの実行/動作設定
0x100	SM2_SHIFTCTRL	ステートマシン2の入出力シフトレジスタの制御動作
0x104	SM2_ADDR	ステートマシン2の現在の命令アドレス
0x108	SM2_INSTR	ステート・マシン2のプログラム・カウンタが現在アドレス指定している命令を確認するための読み出し。 命令を即座に実行し（ジャンプを含む)、その後実行を再開するための書き込み。
0x10c	SM2_PINCTRL	ステートマシン2のピン制御
0x110	SM3_CLKDIV	ステートマシン3用分周レジスタ。 Frequency = clock freq / (CLKDIV_INT + CLKDIV_FRAC / 256)
0x114	SM3_EXECCTRL	ステートマシン3の入出力シフトレジスタの実行/動作設定
0x118	SM3_SHIFTCTRL	ステートマシン3の入出力シフトレジスタの制御動作
0x11c	SM3_ADDR	ステートマシン3の現在の命令アドレス
0x120	SM3_INSTR	ステート・マシン3のプログラム・カウンタが現在アドレス指定している命令を確認するための読み出し。 命令を即座に実行し（ジャンプを含む)、その後実行を再開するための書き込み。
0x124	SM3_PINCTRL	ステートマシン2のピン制御
0x128	RXF0_PUTGET0	Direct read/write access to entry 0 of SM0’s RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set. SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM0 の RX FIFO のエントリ 0 へ直接リード/ライトを行う。
0x12c	RXF0_PUTGET1	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM0 の RX FIFO のエントリ 1 へ直接リード/ライトを行う。
0x130	RXF0_PUTGET2	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM0 の RX FIFO のエントリ 2 へ直接リード/ライトを行う。

0x134	RXF0_PUTGET3	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM0 の RX FIFO のエントリ 3 へ直接リード/ライトを行う。
0x138	RXF1_PUTGET0	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM1 の RX FIFO のエントリ 0 へ直接リード/ライトを行う。
0x13c	RXF1_PUTGET1	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM1 の RX FIFO のエントリ 1 へ直接リード/ライトを行う。
0x140	RXF1_PUTGET2	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM1 の RX FIFO のエントリ 2 へ直接リード/ライトを行う。
0x144	RXF1_PUTGET3	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM1 の RX FIFO のエントリ 3 へ直接リード/ライトを行う。
0x148	RXF2_PUTGET0	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM2 の RX FIFO のエントリ 0 へ直接リード/ライトを行う。
0x14c	RXF2_PUTGET1	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM2 の RX FIFO のエントリ 1 へ直接リード/ライトを行う。
0x150	RXF2_PUTGET2	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM2 の RX FIFO のエントリ 2 へ直接リード/ライトを行う。
0x154	RXF2_PUTGET3	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM2 の RX FIFO のエントリ 3 へ直接リード/ライトを行う。
0x158	RXF3_PUTGET0	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM3 の RX FIFO のエントリ 0 へ直接リード/ライトを行う。
0x15c	RXF3_PUTGET1	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM3 の RX FIFO のエントリ 1 へ直接リード/ライトを行う。
0x160	RXF3_PUTGET2	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM3 の RX FIFO のエントリ 2 へ直接リード/ライトを行う。
0x164	RXF3_PUTGET3	SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET が設定されている場合、SM3 の RX FIFO のエントリ 3 へ直接リード/ライトを行う。
0x168	GPIOBASE	Relocate GPIO 0 (from PIO's point of view) in the system GPIO numbering, to access more than 32 GPIOs from PIO. Only the values 0 and 16 are supported (only bit 4 is writable). PIO から 32 個以上の GPIO にアクセスするために、システム GPIO ナンバリングで（PIO から見て）GPIO 0 を再配置する。値 0 と 16 だけがサポートされています（ビット 4 だけが書き込み可能です）。
0x16c	INTR	Raw Interrupts 生の割込み
0x170	IRQ0_INTE	Interrupt Enable for irq0
0x174	IRQ0_INTF	Interrupt Force for irq0

0x178	IRQ0_INTS	Interrupt status after masking & forcing for irq0
0x17c	IRQ1_INTE	Interrupt Enable for irq1
0x180	IRQ1_INTF	Interrupt Force for irq1
0x184	IRQ1_INTS	Interrupt status after masking & forcing for irq1

Table 981. CTRL Register

31:27	Reserved.	-	-
26	NEXTPREV_CLKDIV_RESTART: Write 1 to restart the clock dividers of state machines in neighbouring PIO blocks, as specified by NEXT_PIO_MASK and PREV_PIO_MASK in the same write. This is equivalent to writing 1 to the corresponding CLKDIV_RESTART bits in those PIOs' CTRL registers.	SC	0x0
25	NEXTPREV_SM_DISABLE: Write 1 to disable state machines in neighbouring PIO blocks, as specified by NEXT_PIO_MASK and PREV_PIO_MASK in the same write. This is equivalent to clearing the corresponding SM_ENABLE bits in those PIOs' CTRL registers.	SC	0x0
24	NEXTPREV_SM_ENABLE: Write 1 to enable state machines in neighbouring PIO blocks, as specified by NEXT_PIO_MASK and PREV_PIO_MASK in the same write. This is equivalent to setting the corresponding SM_ENABLE bits in those PIOs' CTRL registers. If both OTHERS_SM_ENABLE and OTHERS_SM_DISABLE are set, the disable takes precedence.	SC	0x0
23:20	NEXT_PIO_MASK: A mask of state machines in the neighbouring higher-numbered PIO block in the system (or PIO block 0 if this is the highest-numbered PIO block) to which to apply the operations specified by NEXTPREV_CLKDIV_RESTART, NEXTPREV_SM_ENABLE, and NEXTPREV_SM_DISABLE in the same write. This allows state machines in a neighbouring PIO block to be started/stopped/clock-synced exactly simultaneously with a write to this PIO block’s CTRL register. Note that in a system with two PIOs, NEXT_PIO_MASK and PREV_PIO_MASK actually indicate the same PIO block. In this case the effects are applied cumulatively (as though the masks were OR’d together). Neighbouring PIO blocks are disconnected (status signals tied to 0 and control signals ignored) if one block is accessible to NonSecure code, and one is not.	SC	0x0
19:16	PREV_PIO_MASK: A mask of state machines in the neighbouring lower-numbered PIO block in the system (or the highest-numbered PIO block if this is PIO block 0) to which to apply the operations specified by OP_CLKDIV_RESTART, OP_ENABLE, OP_DISABLE in the same write. This allows state machines in a neighbouring PIO block to be started/stopped/clock-synced exactly simultaneously with a write to this PIO block’s CTRL register. Neighbouring PIO blocks are disconnected (status signals tied to 0 and control signals ignored) if one block is accessible to NonSecure code, and one is not.	SC	0x0
15:12	Reserved.	-	-
11:8	CLKDIV_RESTART: Restart a state machine’s clock divider from an initial phase of 0. Clock dividers are free-running, so once started, their output (including fractional jitter) is completely determined by the integer/fractional divisor configured in SMx_CLKDIV. This means that, if multiple clock dividers with the same divisor are restarted simultaneously, by writing multiple 1 bits to this field, the execution clocks of those state machines will run in precise lockstep. Note that setting/clearing SM_ENABLE does not stop the clock divider from		

	running, so once multiple state machines' clocks are synchronised, it is safe to disable/reenable a state machine, whilst keeping the clock dividers in sync. Note also that CLKDIV_RESTART can be written to whilst the state machine is running, and this is useful to resynchronise clock dividers after the divisors (SMx_CLKDIV) have been changed on-the-fly.	SC	0x0
7:4	SM_RESTART: Write 1 to instantly clear internal SM state which may be otherwise difficult to access and will affect future execution. Specifically, the following are cleared: input and output shift counters; the contents of the input shift register; the delay counter; the waiting-on-IRQ state; any stalled instruction written to SMx_INSTR or run by OUT/MOV EXEC; any pin write left asserted due to OUT_STICKY. The contents of the output shift register and the X/Y scratch registers are not affected.	SC	0x0
3:0	SM_ENABLE: Enable/disable each of the four state machines by writing 1/0 to each of these four bits. When disabled, a state machine will cease executing instructions, except those written directly to SMx_INSTR by the system. Multiple bits can be set/cleared at once to run/halt multiple state machines simultaneously.	RW	0x0

PIO: FSTAT Register

Offset: 0x004

Description: FIFO status register

Table 982. FSTAT Register

31:28	Reserved.	-	-
27:24	TXEMPTY: State machine TX FIFO is empty	RO	0xf
23:20	Reserved.	-	-
19:16	TXFULL: State machine TX FIFO is full	RO	0xf
15:12	Reserved.	-	-
11:8	RXEMPTY: State machine RX FIFO is empty	RO	0xf
7:4	Reserved.	-	-
3:0	RXFULL: State machine RX FIFO is full	RO	0xf

PIO: FDEBUG Register

Offset: 0x008

Description: FIFO debug register

Table 983. FDEBUG Register

31:28	Reserved.	-	-
27:24	TXSTALL: State machine has stalled on empty TX FIFO during a blocking WC 0x0 PULL, or an OUT with autopull enabled. Write 1 to clear.	WC	0x0
23:20	Reserved.	-	-
19:16	TXOVER: TX FIFO overflow (i.e. write-on-full by the system) has occurred. WC 0x0 Write 1 to clear. Note that write-on-full does not alter the state or contents of the FIFO in any way, but the data that the system attempted to write is dropped, so if this flag is set, your software has quite likely		

	dropped some data on the floor.	WC	0x0
15:12	Reserved.	-	-
11:8	RXUNDER: RX FIFO underflow (i.e. read-on-empty by the system) has occurred. Write 1 to clear. Note that read-on-empty does not perturb the state of the FIFO in any way, but the data returned by reading from an empty FIFO is undefined, so this flag generally only becomes set due to some kind of software error.	WC	0x0
7:4	Reserved.	-	-
3:0	RXSTALL: State machine has stalled on full RX FIFO during a blocking PUSH, or an IN with autopush enabled. This flag is also set when a nonblocking PUSH to a full FIFO took place, in which case the state machine has dropped data. Write 1 to clear.	WC	0x0

PIO: FLEVEL Register

Offset: 0x00c

Description: FIFO levels

Table 984. FLEVEL Register

31:28	RX3	RO	0x0
27:24	TX3	RO	0x0
23:20	RX2	RO	0x0
19:16	TX2	RO	0x0
15:12	RX1	RO	0x0
11:8	TX1	RO	0x0
7:4	RX0	RO	0x0
3:0	TX0	RO	0x0

PIO: TXF0, TXF1, TXF2, TXF3 Registers

Offsets: 0x010, 0x014, 0x018, 0x01c

Table 985. TXF0, TXF1, TXF2, TXF3 Registers

31:0	Direct write access to the TX FIFO for this state machine. Each write pushes one word to the FIFO. Attempting to write to a full FIFO has no effect on the FIFO state or contents, and sets the sticky FDEBUG_TXOVER error flag for this FIFO.	WF	0x00000000

PIO: RXF0, RXF1, RXF2, RXF3 Registers

Offsets: 0x020, 0x024, 0x028, 0x02c

Table 986. RXF0, RXF1, RXF2, RXF3 Registers

31:0	Direct read access to the RX FIFO for this state machine. Each read pops one word from the FIFO. Attempting to read from an empty FIFO has no effect on the FIFO state, and sets the sticky FDEBUG_RXUNDER error flag		

	for this FIFO. The data returned to the system on a read from an empty FIFO is undefined.	RF	-

PIO: IRQ Register

Offset: 0x030 Bits

Description:

Table 987. IRQ Register

31:8	Reserved.	-	-
7:0	State machine IRQ flags register. Write 1 to clear. There are eight state machine IRQ flags, which can be set, cleared, and waited on by the state machines. There's no fixed association between flags and state machines — any state machine can use any flag. Any of the eight flags can be used for timing synchronisation between state machines, using IRQ and WAIT instructions. Any combination of the eight flags can also routed out to either of the two system-level interrupt requests, alongside FIFO status interrupts — see e.g. IRQ0_INTE.	WC	0x00

PIO: IRQ_FORCE Register

Offset: 0x034

Table 988. IRQ_FORCE Register

31:8	Reserved.	-	-
7:0	Writing a 1 to each of these bits will forcibly assert the corresponding IRQ. Note this is different to the INTF register: writing here affects PIO internal state. INTF just asserts the processor-facing IRQ signal for testing ISRs, and is not visible to the state machines.	WF	0x00

PIO: INPUT_SYNC_BYPASS Register

Offset: 0x038 Bits

Table 989. INPUT_SYNC_BYPASS Register

31:0	There is a 2-flipflop synchronizer on each GPIO input, which protects PIO logic from metastabilities. This increases input delay, and for fast synchronous IO (e.g. SPI) these synchronizers may need to be bypassed. Each bit in this register corresponds to one GPIO. 0 → input is synchronized (default) 1 → synchronizer is bypassed If in doubt, leave this register as all zeroes.	RW	0x00000000

PIO: DBG_PADOUT Register

Offset: 0x03c

Table 990. DBG_PADOUT Register

31:0	Read to sample the pad output values PIO is currently driving to the GPIOs. On RP2040 there are 30 GPIOs, so the two most significant bits are hardwired to 0.	RO	0x00000000

PIO: DBG_PADOE Register

Offset: 0x040

Table 991. DBG_PADOE Register

31:0	Read to sample the pad output enables (direction) PIO is currently driving to the GPIOs. On RP2040 there are 30 GPIOs, so the two most significant bits are hardwired to 0.	RO	0x00000000

PIO: DBG_CFGINFO Register

Offset: 0x044

Description: The PIO hardware has some free parameters that may vary between chip products. These should be provided in the chip datasheet, but are also exposed here.

Table 992. DBG_CFGINFO Register

31:28	VERSION: Version of the core PIO hardware. Enumerated values: 0x0 → V0: Version 0 (RP2040) 0x1 → V1: Version 1 (RP2350)	RO	0x1
27:22	Reserved.	-	-
21:16	IMEM_SIZE: The size of the instruction memory, measured in units of one instruction	RO	-
15:12	Reserved.	-	-
11:8	SM_COUNT: The number of state machines this PIO instance is equipped	RO	-
7:6	Reserved.	RO	-
5:0	with. Reserved. FIFO_DEPTH: The depth of the state machine TX/RX FIFOs, measured in words. Joining fifos via SHIFTCTRL_FJOIN gives one FIFO with double this depth. -	RO	-

PIO: INSTR_MEM0, INSTR_MEM1, …, INSTR_MEM30, INSTR_MEM31 Registers

Offsets: 0x048, 0x04c, …, 0x0c0, 0x0c4

Table 993. INSTR_MEM0, INSTR_MEM1, …, INSTR_MEM30, INSTR_MEM31 Registers

31:16	Reserved.	-	-
15:0	Write-only access to instruction memory location N	WO	0x0000

PIO: SM0_CLKDIV, SM1_CLKDIV, SM2_CLKDIV, SM3_CLKDIV Registers

Offsets: 0x0c8, 0x0e0, 0x0f8, 0x110

Description: Clock divisor register for state machine N Frequency = clock freq / (CLKDIV_INT + CLKDIV_FRAC / 256) Bits Description

Table 994. SM0_CLKDIV, SM1_CLKDIV, SM2_CLKDIV, SM3_CLKDIV Registers

31:16	INT: Effective frequency is sysclk/(int + frac/256). Value of 0 is interpreted as 65536. If INT is 0, FRAC must also be 0.	RW	0x0001
15:8	FRAC: Fractional part of clock divisor	RW	0x00
7:0	Reserved.	-	-

PIO: SM0_EXECCTRL, SM1_EXECCTRL, SM2_EXECCTRL, SM3_EXECCTRL Registers

Offsets: 0x0cc, 0x0e4, 0x0fc, 0x114

Description: Execution/behavioural settings for state machine N

Table 995. SM0_EXECCTRL, SM1_EXECCTRL, SM2_EXECCTRL, SM3_EXECCTRL Registers

31	EXEC_STALLED: If 1, an instruction written to SMx_INSTR is stalled, and latched by the state machine. Will clear to 0 once this instruction completes.	RO	0x0
30	SIDE_EN: If 1, the MSB of the Delay/Side-set instruction field is used as side- set enable, rather than a side-set data bit. This allows instructions to perform side-set optionally, rather than on every instruction, but the maximum possible side-set width is reduced from 5 to 4. Note that the value of PINCTRL_SIDESET_COUNT is inclusive of this enable bit.	RW	0x0
29	SIDE_PINDIR: If 1, side-set data is asserted to pin directions, instead of pin values	RW	0x0
28:24	JMP_PIN: The GPIO number to use as condition for JMP PIN. Unaffected by input mapping.	RW	0x00
23:19	OUT_EN_SEL: Which data bit to use for inline OUT enable	RW	0x00
18	INLINE_OUT_EN: If 1, use a bit of OUT data as an auxiliary write enable When used in conjunction with OUT_STICKY, writes with an enable of 0 will deassert the latest pin write. This can create useful masking/override behaviour due to the priority ordering of state machine pin writes (SM0 < SM1 < …)	RW	0x0
17	OUT_STICKY: Continuously assert the most recent OUT/SET to the pins	RW	0x0
16:12	WRAP_TOP: After reaching this address, execution is wrapped to wrap_bottom. If the instruction is a jump, and the jump condition is true, the jump takes priority.	RW	0x1f
11:7	WRAP_BOTTOM: After reaching wrap_top, execution is wrapped to this address.	RW	0x00
6:5	STATUS_SEL: Comparison used for the MOV x, STATUS instruction. Enumerated values: 0x0 → TXLEVEL: All-ones if TX FIFO level < N, otherwise all-zeroes 0x1 → RXLEVEL: All-ones if RX FIFO level < N, otherwise all-zeroes 0x2 → IRQ: All-ones if the indexed IRQ flag is raised, otherwise all-zeroes	RW	0x0
4:0	STATUS_N: Comparison level or IRQ index for the MOV x, STATUS instruction. If STATUS_SEL is TXLEVEL or RXLEVEL, then values of STATUS_N greater than the current FIFO depth are reserved, and have undefined behaviour. Enumerated values: 0x00 → IRQ: Index 0-7 of an IRQ flag in this PIO block 0x08 → IRQ_PREVPIO: Index 0-7 of an IRQ flag in the next lower-numbered PIO block 0x10 → IRQ_NEXTPIO: Index 0-7 of an IRQ flag in the next higher-numbered PIO block	RW	0x00

PIO: SM0_SHIFTCTRL, SM1_SHIFTCTRL, SM2_SHIFTCTRL, SM3_SHIFTCTRL Registers

Offsets: 0x0d0, 0x0e8, 0x100, 0x118

Description: Control behaviour of the input/output shift registers for state machine N

Table 996. SM0_SHIFTCTRL, SM1_SHIFTCTRL, SM2_SHIFTCTRL, SM3_SHIFTCTRL Registers

31	FJOIN_RX: When 1, RX FIFO steals the TX FIFO’s storage, and becomes twice as deep. TX FIFO is disabled as a result (always reads as both full and empty). FIFOs are flushed when this bit is changed.	RW	0x0
30	FJOIN_TX: When 1, TX FIFO steals the RX FIFO’s storage, and becomes twice as deep. RX FIFO is disabled as a result (always reads as both full and empty). FIFOs are flushed when this bit is changed.	RW	0x0
29:25	PULL_THRESH: Number of bits shifted out of OSR before autopull, or conditional pull (PULL IFEMPTY), will take place. Write 0 for value of 32.	RW	0x00
24:20	PUSH_THRESH: Number of bits shifted into ISR before autopush, or conditional push (PUSH IFFULL), will take place. Write 0 for value of 32.	RW	0x00
19	OUT_SHIFTDIR: 1 = shift out of output shift register to right. 0 = to left.	RW	0x1
18	IN_SHIFTDIR: 1 = shift input shift register to right (data enters from left). 0 = to left.	RW	0x1
17	AUTOPULL: Pull automatically when the output shift register is emptied, i.e. on or following an OUT instruction which causes the output shift counter to reach or exceed PULL_THRESH.	RW	0x0
16	AUTOPUSH: Push automatically when the input shift register is filled, i.e. on an IN instruction which causes the input shift counter to reach or exceed PUSH_THRESH.	RW	0x0
15	FJOIN_RX_PUT: If 1, disable this state machine’s RX FIFO, make its storage available for random write access by the state machine (using the put instruction) and, unless FJOIN_RX_GET is also set, random read access by the processor (through the RXFx_PUTGETy registers). If FJOIN_RX_PUT and FJOIN_RX_GET are both set, then the RX FIFO’s registers can be randomly read/written by the state machine, but are completely inaccessible to the processor. Setting this bit will clear the FJOIN_TX and FJOIN_RX bits.	RW	0x0
14	FJOIN_RX_GET: If 1, disable this state machine’s RX FIFO, make its storage available for random read access by the state machine (using the get instruction) and, unless FJOIN_RX_PUT is also set, random write access by the processor (through the RXFx_PUTGETy registers). If FJOIN_RX_PUT and FJOIN_RX_GET are both set, then the RX FIFO’s registers can be randomly read/written by the state machine, but are completely inaccessible to the processor. Setting this bit will clear the FJOIN_TX and FJOIN_RX bits.	RW	0x0
13:5	Reserved.	-	-
4:0	IN_COUNT: Set the number of pins which are not masked to 0 when read by an IN PINS, WAIT PIN or MOV x, PINS instruction. For example, an IN_COUNT of 5 means that the 5 LSBs of the IN pin group are visible (bits 4:0), but the remaining 27 MSBs are masked to 0. A count of 32 is encoded with a field value of 0, so the default behaviour is to not perform any masking. Note this masking is applied in addition to the masking usually performed by the IN instruction. This is mainly useful for the MOV x,		

	PINS instruction, which otherwise has no way of masking pins.	RW	0x00

PIO: SM0_ADDR, SM1_ADDR, SM2_ADDR, SM3_ADDR Registers

Offsets: 0x0d4, 0x0ec, 0x104, 0x11c

Table 997. SM0_ADDR, SM1_ADDR, SM2_ADDR, SM3_ADDR Registers

31:5	Reserved.	-	-
4:0	Current instruction address of state machine N	RO	0x00

PIO: SM0_INSTR, SM1_INSTR, SM2_INSTR, SM3_INSTR Registers

Offsets: 0x0d8, 0x0f0, 0x108, 0x120

Table 998. SM0_INSTR, SM1_INSTR, SM2_INSTR, SM3_INSTR Registers

31:16	Reserved.	-	-
15:0	Read to see the instruction currently addressed by state machine N's program RW counter. Write to execute an instruction immediately (including jumps) and then resume execution. -	-	-

Table 999. SM0_PINCTRL, SM1_PINCTRL, SM2_PINCTRL, SM3_PINCTRL Registers

31:29	SIDESET_COUNT: The number of MSBs of the Delay/Side-set instruction field which are used for side-set. Inclusive of the enable bit, if present. Minimum of 0 (all delay bits, no side-set) and maximum of 5 (all side-set, no delay).	RW	0x0
28:26	SET_COUNT: The number of pins asserted by a SET. In the range 0 to 5 inclusive.	RW	0x5
25:20	OUT_COUNT: The number of pins asserted by an OUT PINS, OUT PINDIRS or MOV PINS instruction. In the range 0 to 32 inclusive.	RW	0x00
19:15	IN_BASE: The pin which is mapped to the least-significant bit of a state machine's IN data bus. Higher-numbered pins are mapped to consecutively more-significant data bits, with a modulo of 32 applied to pin number.	RW	0x00
14:10	SIDESET_BASE: The lowest-numbered pin that will be affected by a side-set operation. The MSBs of an instruction's side-set/delay field (up to 5, determined by SIDESET_COUNT) are used for side-set data, with the remaining LSBs used for delay. The least-significant bit of the side-set portion is the bit written to this pin, with more-significant bits written to higher-numbered pins.	RW	0x00
9:5	SET_BASE: The lowest-numbered pin that will be affected by a SET PINS or SET PINDIRS instruction. The data written to this pin is the least-significant bit of the SET data.	RW	0x00
4:0	OUT_BASE: The lowest-numbered pin that will be affected by an OUT PINS, RW 0x00 OUT PINDIRS or MOV PINS instruction. The data written to this pin will always be the least-significant bit of the OUT or MOV data.	RW	0x00

PIO: RXF0_PUTGET0 Register

Offset: 0x128

Table 1000. RXF0_PUTGET0 Register

31:0	Direct read/write access to entry 0 of SM0’s RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: SM0_PINCTRL, SM1_PINCTRL, SM2_PINCTRL, SM3_PINCTRL Registers

Offsets: 0x0dc, 0x0f4, 0x10c, 0x124

Description: State machine pin control

31:29	SIDESET_COUNT: The number of MSBs of the Delay/Side-set instruction field which are used for side-set. Inclusive of the enable bit, if present. Minimum of 0 (all delay bits, no side-set) and maximum of 5 (all side-set, no delay).	RW	0x0
28:26	SET_COUNT: The number of pins asserted by a SET. In the range 0 to 5 inclusive.	RW	0x5
25:20	OUT_COUNT: The number of pins asserted by an OUT PINS, OUT PINDIRS or MOV PINS instruction. In the range 0 to 32 inclusive.	RW	0x00
19:15	IN_BASE: The pin which is mapped to the least-significant bit of a state machine’s IN data bus. Higher-numbered pins are mapped to consecutively more-significant data bits, with a modulo of 32 applied to pin number.	RW	0x00
14:10	SIDESET_BASE: The lowest-numbered pin that will be affected by a side-set operation. The MSBs of an instruction’s side-set/delay field (up to 5, determined by SIDESET_COUNT) are used for side-set data, with the remaining LSBs used for delay. The least-significant bit of the side-set portion is the bit written to this pin, with more-significant bits written to higher-numbered pins.	RW	0x00
9:5	SET_BASE: The lowest-numbered pin that will be affected by a SET PINS or SET PINDIRS instruction. The data written to this pin is the least-significant bit of the SET data.	RW	0x00
4:0	OUT_BASE: The lowest-numbered pin that will be affected by an OUT PINS, OUT PINDIRS or MOV PINS instruction. The data written to this pin will always be the least-significant bit of the OUT or MOV data.	RW	0x00

PIO: RXF0_PUTGET1 Register

Offset: 0x12c

Table 1001. RXF0_PUTGET1 Register

31:0	Direct read/write access to entry 1 of SM0’s RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF0_PUTGET2 Register

Offset: 0x130 Bits

Table 1002. RXF0_PUTGET2 Register

31:0	Direct read/write access to entry 2 of SM0's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF0_PUTGET3 Register

Offset: 0x134 Bits

Table 1003. RXF0_PUTGET3 Register

31:0	Direct read/write access to entry 3 of SM0's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF1_PUTGET0 Register

Offset: 0x138 Bits

Table 1004. RXF1_PUTGET0 Register

31:0	Direct read/write access to entry 0 of SM1's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF1_PUTGET1 Register

Offset: 0x13c Bits

Table 1005. RXF1_PUTGET1 Register

31:0	Direct read/write access to entry 1 of SM1's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF1_PUTGET2 Register

Offset: 0x140 Bits

Table 1006. RXF1_PUTGET2 Register

31:0	Direct read/write access to entry 2 of SM1's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF1_PUTGET3 Register

Offset: 0x144

Table 1007. RXF1_PUTGET3 Register

31:0	Direct read/write access to entry 3 of SM1's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF2_PUTGET0 Register

Offset: 0x148 Bits

Table 1008. RXF2_PUTGET0 Register

31:0	Direct read/write access to entry 0 of SM2's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF2_PUTGET1 Register

Offset: 0x14c Bits

Table 1009. RXF2_PUTGET1 Register

31:0	Direct read/write access to entry 1 of SM2's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF2_PUTGET2 Register

Offset: 0x150 Bits

Table 1010. RXF2_PUTGET2 Register

31:0	Direct read/write access to entry 2 of SM2's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF2_PUTGET3 Register

Offset: 0x154

Table 1011. RXF2_PUTGET3 Register

31:0	Direct read/write access to entry 3 of SM2's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF3_PUTGET0 Register

Offset: 0x158 Bits

Table 1012. RXF3_PUTGET0 Register

31:0	Direct read/write access to entry 0 of SM3's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF3_PUTGET1 Register

Offset: 0x15c

Table 1013. RXF3_PUTGET1 Register

31:0	Direct read/write access to entry 1 of SM3's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

Table 1015. RXF3_PUTGET3 Register

PIO: RXF3_PUTGET2 Register

Offset: 0x160 Bits

Table 1014. RXF3_PUTGET2 Register

31:0	Direct read/write access to entry 2 of SM3's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: RXF3_PUTGET3 Register

Offset: 0x164 Bits

Table 1015: RXF3_PUTGET3 Register

31:0	Direct read/write access to entry 3 of SM3's RX FIFO, if SHIFTCTRL_FJOIN_RX_PUT xor SHIFTCTRL_FJOIN_RX_GET is set.	RW	0x00000000

PIO: GPIOBASE Register

Offset: 0x168

Table 1016. GPIOBASE Register

31:5	Reserved.	-	-
4	Relocate GPIO 0 (from PIO's point of view) in the system GPIO numbering, to access more than 32 GPIOs from PIO. Only the values 0 and 16 are supported (only bit 4 is writable).	RW	0x0
3:0	Reserved.	-	-

PIO: INTR Register

Offset: 0x16c

Description: Raw Interrupts

Table 1017. INTR Register

31:16	Reserved.	-	-
15	SM7	RO	0x0

14	SM6	RO	0x0
13	SM5	RO	0x0
12	SM4	RO	0x0
11	SM3	RO	0x0
10	SM2	RO	0x0
9	SM1	RO	0x0
8	SM0	RO	0x0
7	SM3_TXNFULL	RO	0x0
6	SM2_TXNFULL	RO	0x0
5	SM1_TXNFULL	RO	0x0
4	SM0_TXNFULL	RO	0x0
3	SM3_RXNEMPTY	RO	0x0
2	SM2_RXNEMPTY	RO	0x0
1	SM1_RXNEMPTY	RO	0x0
0	SM0_RXNEMPTY	RO	0x0

PIO: IRQ0_INTE Register

Offset: 0x170

Description: Interrupt Enable for irq0 Table 1018. IRQ0_INTE Register

31:16	Reserved.	-	-
15	SM7	RW	0x0
14	SM6	RW	0x0
13	SM5	RW	0x0
12	SM4	RW	0x0
11	SM3	RW	0x0
10	SM2	RW	0x0
9	SM1	RW	0x0
8	SM0	RW	0x0
7	SM3_TXNFULL	RW	0x0
6	SM2_TXNFULL	RW	0x0
5	SM1_TXNFULL	RW	0x0
4	SM0_TXNFULL	RW	0x0
3	SM3_RXNEMPTY	RW	0x0
2	SM2_RXNEMPTY	RW	0x0
1	SM1_RXNEMPTY	RW	0x0
0	SM0_RXNEMPTY	RW	0x0

PIO: IRQ0_INTF Register

Offset: 0x174

Description: Interrupt Force for irq0 Table 1019. IRQ0_INTF Register

31:16	Reserved.	-	-

15	SM7	RW	0x0
14	SM6	RW	0x0
13	SM5	RW	0x0
12	SM4	RW	0x0
11	SM3	RW	0x0
10	SM2	RW	0x0
9	SM1	RW	0x0
8	SM0	RW	0x0
7	SM3_TXNFULL	RW	0x0
6	SM2_TXNFULL	RW	0x0
5	SM1_TXNFULL	RW	0x0
4	SM0_TXNFULL	RW	0x0
3	SM3_RXNEMPTY	RW	0x0
2	SM2_RXNEMPTY	RW	0x0
1	SM1_RXNEMPTY	RW	0x0
0	SM0_RXNEMPTY	RW	0x0

PIO: IRQ0_INTS Register

Offset: 0x178

Description: Interrupt status after masking & forcing for irq0

Table 1020. IRQ0_INTS Register

31:16	Reserved.	–	–
15	SM7	RO	0x0
14	SM6	RO	0x0
13	SM5	RO	0x0
12	SM4	RO	0x0
11	SM3	RO	0x0
10	SM2	RO	0x0
9	SM1	RO	0x0
8	SM0	RO	0x0
7	SM3_TXNFULL	RO	0x0
6	SM2_TXNFULL	RO	0x0
5	SM1_TXNFULL	RO	0x0
4	SM0_TXNFULL	RO	0x0
3	SM3_RXNEMPTY	RO	0x0
2	SM2_RXNEMPTY	RO	0x0
1	SM1_RXNEMPTY	RO	0x0
0	SM0_RXNEMPTY	RO	0x0

PIO: IRQ1_INTE Register

Offset: 0x17c

Description: Interrupt Enable for irq1 Table 1021. IRQ1_INTE Register

31:16	Reserved.	-	-
15	SM7	RW	0x0
14	SM6	RW	0x0
13	SM5	RW	0x0
12	SM4	RW	0x0
11	SM3	RW	0x0
10	SM2	RW	0x0
9	SM1	RW	0x0
8	SM0	RW	0x0
7	SM3_TXNFULL	RW	0x0
6	SM2_TXNFULL	RW	0x0
5	SM1_TXNFULL	RW	0x0
4	SM0_TXNFULL	RW	0x0
3	SM3_RXNEMPTY	RW	0x0
2	SM2_RXNEMPTY	RW	0x0
1	SM1_RXNEMPTY	RW	0x0
0	SM0_RXNEMPTY	RW	0x0

PIO: IRQ1_INTF Register

Offset: 0x180

Description: Interrupt Force for irq1 Table 1022. IRQ1_INTF Register

31:16	Reserved.	-	-
15	SM7	RW	0x0
14	SM6	RW	0x0
13	SM5	RW	0x0
12	SM4	RW	0x0
11	SM3	RW	0x0
10	SM2	RW	0x0
9	SM1	RW	0x0
8	SM0	RW	0x0
7	SM3_TXNFULL	RW	0x0
6	SM2_TXNFULL	RW	0x0
5	SM1_TXNFULL	RW	0x0
4	SM0_TXNFULL	RW	0x0
3	SM3_RXNEMPTY	RW	0x0
2	SM2_RXNEMPTY	RW	0x0
1	SM1_RXNEMPTY	RW	0x0
0	SM0_RXNEMPTY	RW	0x0

PIO: IRQ1_INTS Register

Offset: 0x184

Description: Interrupt status after masking & forcing for irq1 Table 1023. IRQ1_INTS Register

31:16	Reserved.	-	-
15	SM7	RO	0x0
14	SM6	RO	0x0
13	SM5	RO	0x0
12	SM4	RO	0x0
11	SM3	RO	0x0
10	SM2	RO	0x0
9	SM1	RO	0x0
8	SM0	RO	0x0
7	SM3_TXNFULL	RO	0x0
6	SM2_TXNFULL	RO	0x0
5	SM1_TXNFULL	RO	0x0
4	SM0_TXNFULL	RO	0x0
3	SM3_RXNEMPTY	RO	0x0
2	SM2_RXNEMPTY	RO	0x0
1	SM1_RXNEMPTY	RO	0x0
0	SM0_RXNEMPTY	RO	0x0