ANK
ANKL

Logical AND with constant

ANK
ANKL

P851M
P852M
P856M
P857M

Syntax: [label]␣ANK ␣r3, k – T8
[label]␣ANKL␣r1, lk – T2

Logical product

Bit in r3 or r1	Bit in k or lk	Logical product
0	0	0
0	1	0
1	0	0
1	1	1

T8 The logical product of k and the contents of bits 8–15 of the register specified by r3 is placed in bits 8–15 of r3.
Bits 0–7 of this register are set to 0.

T2 The logical product of lk and the contents of the register specified by r1 is placed in r1.

Type	*Function*	*Syntax*
T8	$(r3)_{8-15} \wedge k \rightarrow r3_{8-15}$ $0 \rightarrow r3_{0-7}$	ANK r3, k
T2	$(r1) \wedge lk \rightarrow r1$	ANKL r1, lk

Condition register: CR = 0 if result = 0
1 if result > 0
2 if result < 0

bit	0	1			4	5		7	8							15
T8	0	0	1	0	0	r3			k							

bit	0	1			4	5			8	9	10	11			14	15
T2	1	0	1	0	0	r1				0	1	0	0	0	0	0

Remark:
- * If T8, r3 must be ≠ 0. If T2, r1 must be ≠ 0.
- * Restricted to system mode if r1 = A15.

ANR
ANRS

Logical AND register/register

ANR
ANRS

P851M
P852M
P856M
P857M

Syntax: [label]⌴ ANR [*]⌴ r1, r2
[label]⌴ ANRS ⌴ r1, r2

Logical product

Bit in r1	Bit in 2nd operand	Logical product
0	0	0
0	1	0
1	0	0
1	1	1

The logical product of the contents of the register r1 and the contents of the register specified by r2 (direct addressing) or the contents of the memory address indicated in the register specified by r2 (indirect addressing) is stored in:

- (direct addressing) : register specified by r1
- (indirect addressing) : either in register specified by r1 (I/s = 0) or in the memory address indicated in the register specified by r2 (I/s = 1).

Type	*Function*	*MD*	*I/s*	*Syntax*	
T1	(r1) ∧ (r2) → r1	00	0	ANR	r1, r2
T3	(r1) ∧ ((r2)) → r1	01	0	ANR*	r1, r2
T3	(r1) ∧ ((r2)) → (r2)	01	1	ANRS	r1, r2

Condition register:

CR = 0 if result = 0
1 if result > 0
2 if result < 0

bit 0	1			4	5 8	9 10	11 14	15
1	0	1	0	0	r1	MD	r2	I/s

Remark:
* If T1, then r1 must be ≠ 0. If T3, and I/s ≠ 0 then r1 must be ≠ 0.
* Restricted to system mode if r1 = A15.

AN
ANS

Logical AND

AN
ANS

P851M
P852M
P856M
P857M

Syntax: [label]⌴ AN [S] [*]⌴ r1, m[, r2]

Logical product

Bit in r1	Bit in 2nd operand	Logical product
0	0	0
0	1	0
1	0	0
1	1	1

The logical product of the contents of the effective memory address and the contents of the register specified by r1, is placed in this register, when the I/s indicator 0, or in the effective memory address, when I/s is 1.

Type	*Function*	*MD*	*I/s*	*Syntax*	
T4	(r1) ∧ (m) → r1	10	0	AN	r1, m
T4	(r1) ∧ (m) → m	10	1	ANS	r1, m
T5	(r1) ∧ (m + (r2)) → r1	10	0	AN	r1, m, r2
T5	(r1) ∧ (m + (r2)) → m + (r2)	10	1	ANS	r1, m, r2
T6	(r1) ∧ ((m)) → r1	11	0	AN*	r1, m
T6	(r1) ∧ ((m)) → (m)	11	1	ANS*	r1, m
T7	(r1) ∧ ((m + (r2))) → r1	11	0	AN*	r1, m, r2
T7	(r1) ∧ ((m + (r2))) → (m + (r2))	11	1	ANS*	r1, m, r2

Condition register:

CR = 0 if logical product = 0
1 if logical product > 0
2 if logical product < 0

bit 0	1			4	5 8	9 10	11 14	15
1	0	1	0	0	r1	MD	r2	I/s

Remark:
* If I/s = 0 then r1 must be ≠ 0.
* Restricted to system mode if r1 = A15.

ORK ORKL

Logical OR with constant

ORK ORKL

P851M
P852M
P856M
P857M

Syntax: [label]␣ORK ␣r3, k – T8
[label]␣ORKL␣r1, lk – T2

Logical union:

Bit in r3 or r1	Bit in k or lk	Logical union
0	0	0
0	1	1
1	0	1
1	1	1

T8 A logical OR is performed on the contents of bits 8–15 of the register specified by r3 and the value of the constant k.
The result is placed in bits 8–15 of the register specified by r3.
Bits 0–7 of this register are set to zero.

T2 A logical OR is performed on the contents of the register specified by r1 and the value of the constant lk. The result of this operation is placed in the register specified by r1.

Type	*Function*			*Syntax*
T8	$(r3)_{8-15} \vee k \rightarrow r3_{8-15}$	$r3_{0-7}$	unchanged	ORK r3, k
T2	$(r1) \vee lk \rightarrow r1$			ORKL r1, lk

Condition register:

CR = 0 if result = 0
1 if result > 0
2 if result < 0

bit	0	1 4	5 7	8 15
T8	0	0 1 0 1	r3	k

bit	0	1 4	5 8	9	10	11 14	15
T2	1	0 1 0 1	r1	0	1	0 0 0 0	0

Remark:
- * If I/s = 0 then r1 must be ≠ 0.
- * Restricted to system mode if r1 = A15.

ORR ORRS

Logical OR register/register

ORR ORRS

P851M
P852M
P856M
P857M

Syntax: [label]␣ORR [*]␣r1, r2
[label]␣ORRS␣r1, r2

Logical union:

Bit in r1	Bit in 2nd operand	Logical union
0	0	0
0	1	1
1	0	1
1	1	1

The logical OR of the contents of the 16-bit register specified by r1 and the contents of the 16-bit register specified by r2 (direct addressing) or the contents of the memory address indicated by the register specified by r2 (indirect instruction) is placed:

- (direct addressing) : in the register specified by r1
- (indirect addressing): either in the register specified by r1 (I/s = 0) or in the memory address indicated in the register specified by r2 (I/s = 1).

Type	*Function*	*MD*	*I/s*	*Syntax*	
T1	$(r1) \vee (r2) \rightarrow r1$	00	0	ORR	r1, r2
T3	$(r1) \vee ((r2)) \rightarrow r1$	01	0	ORR*	r1, r2
T3	$(r1) \vee ((r2)) \rightarrow (r2)$	01	1	ORRS	r1, r2

Condition register:

CR = 0 if result =
1 if result > 0
2 if result < 0

bit	0	1 4	5 8	9 10	11 14	15
	1	0 1 0 1	r1	MD	r2	I/s

Remark:
- * If I/s = 0 then r1 must be ≠ 0.
- * Restricted to system mode if r1 = A15.

OR
ORS

Logical OR

OR
ORS

P851M
P852M
P856M
P857M

Syntax: [label]␣ OR[S] [*]␣ r1, m[, r2]

Logical union:

Bit in r1	Bit in 2nd operand	Logical union
0	0	0
0	1	1
1	0	1
1	1	1

The logical OR of the contents of the effective memory address and the contents of the register specified by r1 is placed either in the r1 register, when I/s bit = 0, or in the effective memory address, when I/s bit = 1.

Type	*Function*	*MD*	*I/s*	*Syntax*	
T4	(r1) ∨ (m) → r1	10	0	OR	r1, m
T4	(r1) ∨ (m) → m	10	1	ORS	r1, m
T5	(r1) ∨ (m + (r2)) → r1	10	0	OR	r1, m, r2
T5	(r1) ∨ (m + (r2)) → m + (r2)	10	1	ORS	r1, m, r2
T6	(r1) ∨ ((m)) → r1	11	0	OR*	r1, m
T6	(r1) ∨ ((m)) → (m)	11	1	ORS*	r1, m
T7	(r1) ∨ ((m + (r2))) → r1	11	0	OR*	r1, m, r2
T7	(r1) ∨ ((m + (r2))) → (m + (r2))	11	1	ORS*	r1, m, r2

Condition register:

CR = 0 if result = 0
1 if result > 0
2 if result < 0

bit	0	1 4	5 8	9 10	11 14	15
	1	0 1 0 1	r1	MD	r2	I/s

Remark:
- * r1 must be ≠ 0.
- * Restricted to system mode if r1 = A15.

XRK
XRKL

Exclusive OR with constant

XRK
XRKL

P851M
P852M
P856M
P857M

Syntax:
[label]␣ XRK ␣ r3, k – T8
[label]␣ XRKL␣ r1, lk – T2

Exclusive OR:

Bit in r3 or r1	Bit in k or lk	Exclusive OR
0	0	0
0	1	1
1	0	1
1	1	0

T8 The exclusive OR on the contents of bits 8–15 of the register specified by r3 and the value of k is placed in the register specified by r3.
Bits 0–7 of this register remain unchanged.

T2 The exclusive OR on the contents of the register specified by r1 and lk is placed in the register specified by r1.

Type	*Function*			*Syntax*	
T8	$(r3)_{8-15}$ ∀ k → $r3_{8-15}$	$r3_{0-7}$	unchanged	XRK	r3, k
T2	(r1) ∀ lk → r1			XRK	r1, lk

Condition register:

CR = 0 if result = 0
1 if result > 0
2 if result < 0

bit	0	1 4	5 7	8 15
T8	0	0 1 1 0	r3	k

bit	0	1 4	5 8	9	10	11 14	15
	1	0 1 1 0	r1	0	1	0 0 0 0	0

Remark:
- • r1 and r3 must be ≠ 0.
- • Restricted to system mode if r1 = A15.

XR
XRS

Exclusive OR

XR
XRS

P851M
P852M
P856M
P857M

Syntax: [label]⌴ XR[S] [*]⌴ r1, m[, r2]

Exclusive OR:

Bit in r1	Bit in 2nd operand	Exclusive OR
0	0	0
0	1	1
1	0	1
1	1	0

The exclusive OR of the contents of the effective memory address and the contents of the register specified by r1 is placed either in the register specified by r1, when the I/s bit = 0 or in the effective memory address, when I/s = 1.

Type	*Function*	*MD*	*I/s*	*Syntax*	
T4	(r1) ∀ (m) → r1	10	0	XR	r1, m
T4	(r1) ∀ (m) → m	10	1	XRS	r1, m
T5	(r1) ∀ (m + (r2)) → r1	10	0	XR	r1, m, r2
T5	(r1) ∀ (m + (r2)) → m + (r2)	10	1	XRS	r1, m, r2
T6	(r1) ∀ ((m)) → r1	11	0	XR*	r1, m
T6	(r1) ∀ ((m)) → (m)	11	1	XRS*	r1, m
T7	(r1) ∀ ((m + (r2))) → r1	11	0	XR*	r1, m, r2
T7	(r1) ∀ ((m + (r2))) → (m + (r2))	11	1	XRS*	r1, m, r2

Condition register:

CR = 0 if result = 0
1 if result > 0
2 if result < 0

bit 0	1 4	5 8	9 10	11 14	15
1	0 1 1 0	r1	MD	r2	I/s

Remark:
* r1 must be ≠ 0.
* Restricted to system mode if r1 = A15.

XRR
XRRS

Exclusive OR register/register

XRR
XRRS

P851M
P852M
P856M
P857M

Syntax: [label]⌴ XRR [*]⌴ r1, r2
[label]⌴ XRRS ⌴ r1, r2

Exclusive OR:

Bit in r1	Bit in 2nd operand	Exclusive OR
0	0	0
0	1	1
1	0	1
1	1	0

The exclusive OR of the contents of the 16-bit register specified by r1 and the contents of the 16-bit register specified by r2 (direct addressing) or the contents of the memory address indicated in the register specified by r2 (indirect addressing) are placed as follows:

- (direct addressing) : in the register specified by r1
- (indirect addressing) : either in the register specified by r1 (I/s = 0) or in the memory address indicated by the register specified by r2 (I/s = 1).

Type	*Function*	*MD*	*I/s*	*Syntax*	
T1	(r1) ∀ (r2) → r1	00	0	XRR	r1, r2
T3	(r1) ∀ ((r2)) → r1	01	0	XRR*	r1, r2
T3	(r1) ∀ ((r2)) → (r2)	01	1	XRRS	r1, r2

Condition register:

CR = 0 if result = 0
1 if result > 0
2 if result < 0 .

bit 0	1 4	5 8	9 10	11 14	15
1	0 1 1 0	r1	MD	r2	I/s

Remark:
* r1 must be ≠ 0.
* Restricted to system mode if r1 = A15.

TM

Test mask

TM

P851M
P852M
P856M
P857M

Syntax: [label]␣TM␣r1, r2

The logical product (AND) of the contents of the register specified by r1 and the contents of the register specified by r2 is compared to zero. The result of the comparison is stored in the condition register. The contents of the register specified by r1 and r2 remain unchanged.

Type	*Function*
T1	$[(r1) \wedge (r2)] \leftrightarrow 0 \rightarrow CR$

Condition register: CR = 0 if result = 0
1 if result > 0
2 if result < 0

bit	0	1			4	5	8	9	10	11	14	15
	1	0	1	0	0	r1		0	0	r2		1

Remark:
- r1 must be ≠ 0.
- Restricted to system mode if r1 = A15.

TNM

Test not mask

TNM

P851M
P852M
P856M
P857M

Syntax: [label]␣TNM␣r1, r2

The exclusive OR of the contents of the register specified by r1 and the contents of the register specified by r2 is compared with zero. The result of the comparison is stored in the condition register. The initial contents of the register specified by r1 and the register specified by r2 remain unchanged.

Type	*Function*
T1	$[(r1) \veebar (r2)] \leftrightarrow 0 \rightarrow CR$

Condition register: CR = 0 if result = 0
1 if result > 0
2 if result < 0

bit	0	1			4	5	8	9	10	11	14	15
	1	0	1	1	0	r1		0	0	r2		1

Remark:
- r1 must be ≠ 0.
- Restricted to system mode if r1 = A15.

TSB Test and Set Bit TSB

P853
P854
P858
P859

Syntax: [label] TSB[*] m[,r2]

This instruction tests a bit in a bitstring, sets the condition register to the value of that bit, and sets the bit to 1.
The address of the first character of the bitstring is the instruction operand, found as follows:

Type	*address*
T4	m
T5	m + (r2)
T6	(m)
T7	(m + (r2))

The bit position in the string must be specified in register A2; in addressing the operand it is used as shown below:

	0 12	13 15
A2	D	B

The bit displacement $A2_{0-15}$ is split up in the character displacement D and the bit number B.

The function of the instruction is:

Type	*Function*	*Mode*	*Syntax*	
T4	$(m + D)_B \rightarrow CR$ $1 \rightarrow (m + D)_B$	10	TSB	m
T5	$(m + (r2) + D)_B \rightarrow CR$ $1 \rightarrow (m + (r2) + D)_B$	10	TSB	m,r2
T6	$((m) + D)_B \rightarrow CR$ $1 \rightarrow ((m) + D)_B$	11	TSB*	m
T7	$((m + (r2)) + D)_B \rightarrow CR$ $1 \rightarrow ((m + (r2)) + D)_B$	11	TSB*	m,r2

Condition register:

CR = 0 if tested bit was 0
CR = 1 if tested bit was 1

bit	0	1 4	5 8	9 10	11 14	15
	1	1 0 0 0	0 0 0 0	mode	r2	1

TSBR Test and Test Bit / Register TSBR

P853
P854
P858
P859

Syntax: [label] TSBR r2

This instruction tests a bit in a bitstring, sets the condition register to the value of that bit, and sets the bit to 1.
The address of the first character of the string is contained in the register specified by r2.
The bit position in the string must be specified in register A2; in addressing the operand it is used as shown below:

	0 12	13 15
A2	D	B

The bit displacement $A2_{0-15}$ is split up in the character displacement D and the bit number B.

The function of the instruction is:

Type	*Function*
T3	$((r2) + D)_B \rightarrow CR$ $1 \rightarrow ((r2) + D)_B$

Condition register:

CR = 0 if the tested bit was 0
CR = 1 if the tested bit was 1

bit	0	1 4	5 8	9 10	11 14	15
	1	1 0 0 0	0 0 0 0	0 1	r2	1

TRB — Test and Reset Bit — TRB

P853
P854
P858
P859

Syntax: [label] TRB[*] m[,r2]

This instruction tests a bit in a bitstring, sets the condition register to the value of that bit, and resets the bit to 0.
The address of the first character of the bitstring is the instruction operand, found as follows:

Type	*Address*
T4	m
T5	m + (r2)
T6	(m)
T7	(m + (r2))

The bit position in the string must be specified in register A2; in addressing the operand it is used as shown below:

	0	12	13	15
A2	D		B	

The bit displacement $A2_{0-15}$ is split up in the character displacement D and the bit number B.

The function of the instruction is:

Type	*Function*	*Mode*	*Syntax*	
T4	$(m + D)_B \rightarrow CR$ $0 \rightarrow (m + D)_B$	10	TRB	m
T5	$(m + (r2) + D)_B \rightarrow CR$ $0 \rightarrow (m + (r2) + D)_B$	10	TRB	m,r2
T6	$((m) + D)_B \rightarrow CR$ $0 \rightarrow ((m) + D)_B$	11	TRB*	m
T7	$((m + (r2)) + D)_B \rightarrow CR$ $0 \rightarrow ((m + (r2)) + D)_B$	11	TRB*	m,r2

Condition register:

CR = 0 if tested bit was 0
CR = 1 if tested bit was 1

bit	0	1 4	5 8	9 10	11 14	15
	1	1 0 0 1	0 0 0 0	mode	r2	1

TRBR — Test and Reset Bit / Register — TRBR

P853
P854
P858
P859

Syntax: [label] TRBR r2

This instruction tests a bit in a bitstring, sets the condition register to the value of that bit, and resets the bit to 0.
The address of the first character of the string is contained in the register specified by r2.
The bit position in the string must be specified in register A2; in addressing the operand it is used as shown below:

	0	12	13	15
A2	D		B	

The bit displacement $A2_{0-15}$ is split up in the character displacement D and the bit number B.

The function of the instruction is:

Type	*Function*
T3	$((r2) + D)_B \rightarrow CR$ $0 \rightarrow ((r2) + D)_B$

Condition register:

CR = 0 if the tested bit was 0
CR = 1 if the tested bit was 1

bit	0	1 4	5 8	9 10	11 14	15
	1	1 0 0 1	0 0 0 0	0 1	r2	1

TB

Test Bit

TB

P853
P854
P858
P859

Syntax: [label] TB[*] m[,r2]

This instruction tests a bit in a bitstring, and sets the condition register to the value of that bit.
The address of the first character of the bitstring is the instruction operand, found as follows:

Type	*Address*
T4	m
T5	m + (r2)
T6	(m)
T7	(m + (r2))

The bit position in the string must be specified in register A2; in addressing the operand it is used as shown below:

	0 … 12	13 … 15
A2	D	B

The bit displacement $A2_{0-15}$ is split in the character displacement D and the bit number B.

The function of the instruction is:

Type	*Function*	*Mode*	*Syntax*	
T4	$(m + D)_B \rightarrow CR$	10	TB	m
T5	$(m + (r2) + D)_B \rightarrow CR$	10	TB	m,r2
T6	$((m) + D)_B \rightarrow CR$	11	TB*	m
T7	$((m + (r2)) + D)_B \rightarrow CR$	11	TB*	m,r2

Condition register:

CR = 0 if tested bit was 0
CR = 1 if tested bit was 1

bit	0	1 … 4	5 … 8	9 10	11 … 14	15
	1	1 0 1 0	0 0 0 0	mode	r2	1

TBR

Test Bit / Register

TBR

P853
P854
P858
P859

Syntax: [label] TBR r2

This instruction tests a bit in a bitstring, sets the condition register to the value of that bit.
The address of the first character of the string is contained in the register specified by r2.
The bit position in the string must be specified in register A2; in addressing the operand it is used as shown below:

	0 … 12	13 … 15
A2	D	B

The bit displacement $A2_{0-15}$ is split up in the character displacement D and the bit number B.

The function of the instruction is:

Type	*Function*
T3	$((r2) + D)_B \rightarrow CR$

Condition register:

CR = 0 if the tested bit was 0
CR = 1 if the tested bit was 1

bit	0	1 … 4	5 … 8	9 10	11 … 14	15
	1	1 0 1 0	0 0 0 0	0 1	r2	1