MACHINE
PIC

SEES
BIT_DEFINITION,
BYTE_DEFINITION,
TYPES

USES
ALU

VARIABLES
mem,
W_REGISTER,
pc,
stack,
sp

INVARIANT
$mem \in REGISTER \rightarrow BYTE \wedge$
$W\_REGISTER \in BYTE \wedge$
$pc \in INSTRUCTION \wedge$
$sp \in \mathcal{N} \wedge$
$stack \in \mathcal{N} \nrightarrow INSTRUCTION \wedge$
$\mathbf{dom}(stack) = 0 \mathinner{..} (sp\text{-}1)$

CONSTANTS

IRP_POS, *RP1_POS*, *RP0_POS*, *TO_POS*, *PD_POS*,
ZERO_POS, *DCARRY_POS*, *CARRY_POS*,

INDF_ADDR0, *TMR0_ADDR*, *PCL_ADDR*,
STATUS_ADDR0, *FSR_ADDR0*,
PORTA_ADDR, *PORTB_ADDR*,
PCLATH_ADDR0, *INTCON_ADDR0*, *PIR1_ADDR*, *CMCON_ADDR*,
INDF_ADDR1, *STATUS_ADDR1*, *FSR_ADDR1*,
TRISA_ADDR, *TRISB_ADDR*,
PCLATH_ADDR1, *INTCON_ADDR1*, *PIE1_ADDR*, *PCON_ADDR*, *LININTF_ADDR*, *VRCON_ADDR*,

UNIMPLEMENTED_LOCATIONS,

get_address,

get_offset

PROPERTIES

$INDF\_ADDR0 \in REGISTER \wedge \quad INDF\_ADDR0 = 0 \wedge$
$TMR0\_ADDR \in REGISTER \wedge \quad TMR0\_ADDR = 1 \wedge$
$PCL\_ADDR \in REGISTER \wedge \quad PCL\_ADDR = 2 \wedge$
$STATUS\_ADDR0 \in REGISTER \wedge \quad STATUS\_ADDR0 = 3 \wedge$
$FSR\_ADDR0 \in REGISTER \wedge \quad FSR\_ADDR0 = 4 \wedge$
$PORTA\_ADDR \in REGISTER \wedge \quad PORTA\_ADDR = 5 \wedge$
$PORTB\_ADDR \in REGISTER \wedge \quad PORTB\_ADDR = 6 \wedge$
$PCLATH\_ADDR0 \in REGISTER \wedge \quad PCLATH\_ADDR0 = 10 \wedge$
$INTCON\_ADDR0 \in REGISTER \wedge \quad INTCON\_ADDR0 = 11 \wedge$
$PIR1\_ADDR \in REGISTER \wedge \quad PIR1\_ADDR = 12 \wedge$
$CMCON\_ADDR \in REGISTER \wedge \quad CMCON\_ADDR = 127 \wedge$
$INDF\_ADDR1 \in REGISTER \wedge \quad INDF\_ADDR1 = 128 \wedge$
$STATUS\_ADDR1 \in REGISTER \wedge \quad STATUS\_ADDR1 = 131 \wedge$
$FSR\_ADDR1 \in REGISTER \wedge \quad FSR\_ADDR1 = 132 \wedge$
$TRISA\_ADDR \in REGISTER \wedge \quad TRISA\_ADDR = 133 \wedge$
$TRISB\_ADDR \in REGISTER \wedge \quad TRISB\_ADDR = 134 \wedge$
$PCLATH\_ADDR1 \in REGISTER \wedge \quad PCLATH\_ADDR1 = 138 \wedge$
$INTCON\_ADDR1 \in REGISTER \wedge \quad INTCON\_ADDR1 = 139 \wedge$
$PIE1\_ADDR \in REGISTER \wedge \quad PIE1\_ADDR = 140 \wedge$
$PCON\_ADDR \in REGISTER \wedge \quad PCON\_ADDR = 142 \wedge$
$LININTF\_ADDR \in REGISTER \wedge \quad LININTF\_ADDR = 144 \wedge$
$VRCON\_ADDR \in REGISTER \wedge \quad VRCON\_ADDR = 159 \wedge$

$UNIMPLEMENTED\_LOCATIONS =$
$\quad 7 \mathinner{..} 9 \cup \quad 13 \mathinner{..} 30 \cup 135 \mathinner{..} 137 \cup \{141\} \cup \{143\} \cup 145 \mathinner{..} 158 \cup 192 \mathinner{..} 240 \wedge$

$IRP\_POS \in BYTE\_INDEX \wedge \quad IRP\_POS = 7 \wedge$
$RP1\_POS \in BYTE\_INDEX \wedge \quad RP1\_POS = 6 \wedge$
$RP0\_POS \in BYTE\_INDEX \wedge \quad RP0\_POS = 5 \wedge$
$TO\_POS \in BYTE\_INDEX \wedge \quad TO\_POS = 4 \wedge$
$PD\_POS \in BYTE\_INDEX \wedge \quad PD\_POS = 3 \wedge$
$ZERO\_POS \in BYTE\_INDEX \wedge \quad ZERO\_POS = 2 \wedge$
$DCARRY\_POS \in BYTE\_INDEX \wedge \quad DCARRY\_POS = 1 \wedge$
$CARRY\_POS \in BYTE\_INDEX \wedge \quad CARRY\_POS = 0 \wedge$

$get\_offset : ((REGISTER \times REGISTER) \rightarrow REGISTER) \wedge$
$\forall (ff,ind).\ (ff \in REGISTER \wedge ind \in REGISTER \Rightarrow$
$\quad ((ff = INDF\_ADDR0 \Rightarrow get\_offset(ind,ff) = ind) \wedge$
$\quad (ff \neq INDF\_ADDR0 \Rightarrow get\_offset(ind,ff) = ff))) \wedge$
$get\_address : ((REGISTER \times BIT \times REGISTER) \rightarrow REGISTER) \wedge$
$\forall (ff,bb,ind).\ (ff \in REGISTER \wedge bb \in BIT \wedge ind \in REGISTER \Rightarrow$
$\quad (get\_address(ff,bb,ind) = 128 \times bb + get\_offset(ff,ind)))$

ASSERTIONS

ran$(mem) \subseteq BYTE$;
dom$(mem) = REGISTER$

DEFINITIONS

$INDF\_VAL0 == mem(INDF\_ADDR0)$;
$TMR0\_VAL == mem(TMR0\_ADDR)$;
$PCL\_VAL == mem(PCL\_ADDR)$;
$STATUS\_VAL0 == mem(STATUS\_ADDR0)$;
$FSR\_VAL0 == mem(FSR\_ADDR0)$;
$FSR\_VAL == FSR\_VAL0$;
$INDF\_REGISTER == INDF\_VAL0$;
$STATUS\_REGISTER == STATUS\_VAL0$;
$PORTA\_VAL == mem(PORTA\_ADDR)$;
$PORTB\_VAL == mem(PORTB\_ADDR)$;
$INDF\_VAL1 == mem(INDF\_ADDR1)$;
$STATUS\_VAL1 == mem(STATUS\_ADDR1)$;
$FSR\_VAL1 == mem(FSR\_ADDR1)$;
$TRISA\_VAL == mem(TRISA\_ADDR)$;
$TRISB\_VAL == mem(TRISB\_ADDR)$;

$RP1\_VAL0 == STATUS\_VAL0(RP1\_POS)$;
$RP1\_VAL1 == STATUS\_VAL1(RP1\_POS)$;

$RP0\_VAL0 == STATUS\_VAL0(RP0\_POS)$;
$RP0\_VAL1 == STATUS\_VAL1(RP0\_POS)$;
$RP0\_VAL == RP0\_VAL0$;

$ZERO\_VAL0 == STATUS\_VAL0(ZERO\_POS)$;
$ZERO\_VAL1 == STATUS\_VAL1(ZERO\_POS)$;

$DCARRY\_VAL0 == STATUS\_VAL0(DCARRY\_POS)$;
$DCARRY\_VAL1 == STATUS\_VAL1(DCARRY\_POS)$;

$CARRY\_VAL0 == STATUS\_VAL0(CARRY\_POS)$;
$CARRY\_VAL1 == STATUS\_VAL1(CARRY\_POS)$;

$update\_status\_register(zero, digit\_carry, carry) ==$
$(STATUS\_REGISTER \leftarrow\!\!\!\!\triangleleft \{ ZERO\_POS \mapsto bool\_to\_bit(zero),$
$DCARRY\_POS \mapsto bool\_to\_bit(digit\_carry),$
$CARRY\_POS \mapsto bool\_to\_bit(carry)\})$;
$update\_status(zero, digit\_carry, carry) ==$
$(mem \leftarrow\!\!\!\!\triangleleft \{ STATUS\_ADDR0 \mapsto update\_status\_register(zero, digit\_carry, carry),$
$STATUS\_ADDR1 \mapsto update\_status\_register(zero, digit\_carry, carry) \})$

INITIALISATION
$W\_REGISTER :\in BYTE \;||$
ANY *random*, *init_status* **WHERE**
$random \in REGISTER \rightarrow BYTE \wedge$
$init\_status \in BYTE \wedge$
$init\_status\ (RP1\_POS) = 0$
THEN
$mem := random \leftarrow\!\!\!\!\triangleleft \{ INDF\_ADDR0 \mapsto BYTE\_ZERO,$
$STATUS\_ADDR0 \mapsto init\_status,$
$INDF\_ADDR0 \mapsto BYTE\_ZERO,$
$STATUS\_ADDR1 \mapsto init\_status \}$

$\lhd$ (*UNIMPLEMENTED_LOCATIONS* $\times$ { *BYTE_ZERO* })

END ||

pc :$\in$ *INSTRUCTION* ||

stack := $\emptyset$ ||

sp := 0

OPERATIONS

CALL (*kk*) =

PRE *kk* $\in$ *INSTRUCTION* **THEN**

stack(*sp*) := *instruction_next*(*pc*) ||

sp := *sp* + 1 ||

pc := *kk*

END

;

RETURN =

PRE *sp* $>$ 0 **THEN**

stack := {*sp* - 1} $\lhd$ *stack* ||

pc := *stack*(*sp* - 1) ||

sp := *sp* - 1

END

;

RETLW (*kk*) =

PRE *kk* $\in$ *UCHAR* $\wedge$ *sp* $>$ 0 **THEN**

pc := *stack*(*sp* - 1) ||

W_REGISTER := *uchar_to_byte*(*kk*)

END

;

ADDLW (*kk*) =

PRE *kk* $\in$ *UCHAR* **THEN**

ANY *result*, *carry*, *digit_carry*, *zero* **WHERE**

result $\in$ *UCHAR* $\wedge$ *digit_carry* $\in$ **BOOL** $\wedge$ *carry* $\in$ **BOOL** $\wedge$ *zero* $\in$ **BOOL** $\wedge$

result, *digit_carry*, *carry*, *zero* = *add*(*kk*, *byte_to_uchar*(*W_REGISTER*))

THEN

W_REGISTER := *uchar_to_byte*(*result*) ||

mem := *update_status*(*zero*, *digit_carry*, *carry*)

END ||

pc := *instruction_next*(*pc*)

END

;

ADDWF (*ff*, *dd*) =

PRE *ff* $\in$ *REGISTER0* $\wedge$ *dd* $\in$ *BIT* **THEN**

ANY *dest*, *result*, *carry*, *digit_carry*, *zero* **WHERE**

dest $\in$ *REGISTER* $\wedge$

result $\in$ *UCHAR* $\wedge$ *digit_carry* $\in$ **BOOL** $\wedge$ *carry* $\in$ **BOOL** $\wedge$ *zero* $\in$ **BOOL** $\wedge$

dest = *get_address*(*ff*, *RP0_VAL*, *byte_to_uchar*(*FSR_VAL*)) $\wedge$

result, *digit_carry*, *carry*, *zero* = *add*(*byte_to_uchar*(*mem*(*dest*)), *byte_to_uchar*(*W_REGISTER*))

THEN

ANY *mem2*, *bv* **WHERE**

bv $\in$ *BYTE* $\wedge$ *bv* = *uchar_to_byte*(*result*) $\wedge$

mem2 $\in$ *REGISTER* $\rightarrow$ *BYTE* $\wedge$

(*dd* $\neq$ 0 $\wedge$ *dest* $\notin$ *UNIMPLEMENTED_LOCATIONS* $\Rightarrow$ *mem2* = *mem* $\lhd$ { *dest* $\mapsto$ *bv* }) $\wedge$

```
            (dd = 0 ∨ dest ∈ UNIMPLEMENTED_LOCATIONS ⇒ mem2 = mem)
          THEN
            IF dd = 0 THEN
              W_REGISTER := bv
            END ||
            IF ff ≠ STATUS_ADDR0 THEN
              mem := update_status(zero, digit_carry, carry)
            END
          END
        END ||
        pc := instruction_next(pc)
      END
END
```