ORG 0x100 ; variables & masks

INDEX: WORD 0x06D6 ; current pointer into array (auto-++ by (PTR)+)

COUNT: WORD 0x000E ; 14 elements in total

I: WORD 0x0000 ; loop counter 0…13 (parity test)

FIRST: WORD 0x0001 ; 1 = max not initialised yet

W0: WORD 0x0000 ; low word of current element

W1: WORD 0x0000 ; high word (after sign-extend)

TMP: WORD 0x0000 ; temp for compares / flags

mask: word 0x8000

SIGN\_M: WORD 0x0002 ; bit-17 mask inside high word

POS\_M: WORD 0x0003 ; keep exactly two useful high bits for + numbers

NEG\_M: WORD 0xFFFC ; fill upper 14 bits with 1 for –

MAX0: WORD 0x0000 ; current max: low

MAX1: WORD 0x0000 ; current max: high

; ---------------------------------------------------------------------

ORG 0x120 ; code

START: CLA

ST I ; I = 0

MAIN\_L: LD I

CMP COUNT

BZS FINISH ; done when I == COUNT

; ---- read element (Lo then Hi) --------------------------------

LD (INDEX)+

ST W0

LD (INDEX)+

ST W1

; ---- 18-bit sign-extension to full 16-bit Hi word -------------

LD W1

AND SIGN\_M

BZS POS\_VAL ; sign bit = 0 → +

NEG\_VAL:LD W1

OR NEG\_M ; fill with 1s

ST W1

JUMP EXT\_DONE

POS\_VAL:LD W1

AND POS\_M

ST W1

EXT\_DONE:

; ---- check odd logical index

LD I

AND #0x0001

BZS SKIP\_ELEM ; even → ignore

; ---- first qualifying element becomes the initial MAX

LD FIRST

BZS COMPARE ; FIRST == 0 ? → already have max

LD W0

ST MAX0

LD W1

ST MAX1

CLA

ST FIRST

JUMP NEXT\_I

COMPARE: ; ---- compare candidate (W1:W0) with current MAX

; high words first (signed):

LD W1

SUB MAX1

ST TMP ; keep the diff

AND $mask

BZS HIGH\_GE ; sign of diff == 0 → cand\_hi ≥ max\_hi

JUMP NEXT\_I ; cand\_hi < max\_hi → skip update

HIGH\_GE:

LD TMP

BZS HIGH\_EQ ; diff == 0 ?

; cand\_hi > max\_hi

UPDATE: LD W0

ST MAX0

LD W1

ST MAX1

JUMP NEXT\_I

HIGH\_EQ: ; --- same high word, compare lows (unsigned is enough here)

LD W0

SUB MAX0

ST TMP

AND $mask

BZS LOW\_GE ; cand\_lo ≥ max\_lo ?

JUMP NEXT\_I ; smaller → skip

LOW\_GE: LD TMP

BZS NEXT\_I ; equal → keep old max

; cand\_lo > max\_lo → update

LD W0

ST MAX0

LD W1

ST MAX1

NEXT\_I: ;advance loop counters

LD I

INC

ST I

JUMP MAIN\_L

SKIP\_ELEM:

LD I

INC

ST I

JUMP MAIN\_L

FINISH: ; write 32-bit result to 0x400 and hlt

LD MAX0

ST $RES0

LD MAX1

ST $RES1

HLT

; result:

ORG 0x400

RES0: WORD 0x0000 ; low word of MAX

RES1: WORD 0x0000 ; high word of MAX

; TEST DATA (14 elements × 2 words)

; ---------------------------------------------------------------------

ORG 0x6D6

ARRAY: WORD 0x1111, 0x0000 ; index 4 (even) – ignored

WORD 0xFFaF, 0x0002 ; index 5 (odd) – candidate 0001 ffaf

WORD 0xFFFF, 0x0002 ; index 6 (even)

WORD 0xFFFF, 0x0002 ; index 7 (odd) - candidate

WORD 0x2222, 0x0000

WORD 0x3333, 0x0000

WORD 0x4444, 0x0000

WORD 0x5555, 0x0000

WORD 0x6666, 0x0000

WORD 0x7777, 0x0000

WORD 0x8888, 0x0000

WORD 0x9999, 0x0000

WORD 0xAAAA, 0x0000

WORD 0xBBBB, 0x0000