XDEF switch\_file

XREF delayon, startuptimer, delaytimer, switchchange

XREF switchstatus, prevswitchstatus, port\_t, disp5, display\_string

XREF ledswitches, switchchecker, hexkeypad, wait, dispc, disp3

XREF dispF, dispe, holdold, gen1off, gen2off, gen3off, dispd

XREF waithold, disp2, noTRACKER, rows, lookup, PTU, TRACKER, HOMEflg

XREF sum, value, TON, real\_power\_output , real\_value, pot\_value, power\_output, read\_pot, shutoff

XREF shutdown, switchflg, gen1cap, gen2cap, gen3cap, times8, disp

XREF home1, dispa, HOME, sendhome, port\_s, GenPick, disp9, time1off, time2off, time3off

XREF g1a, g2a, g3a, autoshut1, autoshut2, autoshut3, gen1au, gen2au, gen3au

;This file is for cheching if the switches have been changed

; I saves the value of the switches and saves them to compare to new values

; If it ever changes then you have to type in your pass word correclty

; If you do not then it tells you to reset them

; It is important to remeber to mask the fourth and fith switch because they are used to do differetn things

; It is also important to note that switch 8 does note trigger this to happen

switch\_file:

MOVB #0, HOMEflg ; let rest of program know its not at home

MOVB #0, noTRACKER ; don’t exit keypad prematurely

PSHX

PSHY

PSHD

MOVB #1, delayon

MOVB #0, delaytimer

MOVB switchstatus, prevswitchstatus ; save previous switch status

switchdelay1:LDAA delaytimer

CMPA #1 ; add small delay so program can allow for

; multiple switch adjustments at one time

BNE switchdelay1

MOVB #0, switchchange ; let rest of program know switch change is

; accounted for

MOVB #0, delayon

LDAA port\_t

ANDA #$87

STAA switchstatus ; save new switch status

LDAA switchstatus

ANDA #$80

LDAB prevswitchstatus

ANDB #$80

STAB switchchecker

CMPA switchchecker

BEQ not8 ; check if auto shutoff switch was changed

LDAA times8

INCA ; need to flip auto shutoff switch twice so keep

; track of that through times8 variable

STAA times8

CMPA #2

BEQ autor

PULD

PULY

PULX

RTS

autor: JSR auto

PULD

PULY

PULX

RTS

nochange1aa: MOVB #0, gen1au ; allow normal switch functions to begin again

LDAA switchstatus

ANDA #1 ; check if generator 1 is on

CMPA #1

BEQ disr1

BRA nochange1

nochange2aa: MOVB #0, gen2au

LDAA switchstatus

ANDA #2

CMPA #2

BEQ disr2

BRA nochange2

nochange3aa: MOVB #0, gen3au

LDAA switchstatus

ANDA #4

CMPA #4

BEQ disr3

JMP nochange3

not8:

LDAA switchstatus

ANDA #1

LDAB prevswitchstatus

ANDB #1

STAB switchchecker ; check if switch 1(generator 1) was changed

CMPA switchchecker

BEQ nochange1

LDAA gen1au ; check if generator 1 went through autoshutoff

CMPA #1

BEQ nochange1aa

disr1: LDAA ledswitches ; make sure homescreen indicated generator 1 is on

ORAA #1

STAA ledswitches

BRA skip01

nochange1:LDAA ledswitches

ORAA #0 ; make sure homescreen indicates generator 1 is off

BCLR ledswitches, #1

skip01: LDAA switchstatus ; repeat above generator 1 process for 2 and 3. Refer to

; generator 1 comments for details

ANDA #2

LDAB prevswitchstatus

ANDB #2

STAB switchchecker

CMPA switchchecker

BEQ nochange2

LDAA gen2au

CMPA #1

BEQ nochange2aa

disr2: LDAA ledswitches

ORAA #2

STAA ledswitches

BRA skip02

nochange2: LDAA ledswitches

BCLR ledswitches, #2

skip02: LDAA switchstatus

LDAA switchstatus

ANDA #4

LDAB prevswitchstatus

ANDB #4

STAB switchchecker

CMPA switchchecker

BEQ nochange3

LDAA gen3au

CMPA #1

BEQ nochange3aajmp

BRA disr3

nochange3aajmp: JMP nochange3aa

disr3: LDAA ledswitches

ORAA #4

STAA ledswitches

BRA skip03

nochange3: LDAA ledswitches

BCLR ledswitches, #4

skip03: LDAA ledswitches

CMPA #0

BNE dontleave

PULD

PULY

PULX

RTS

dontleave: LDAA switchstatus

MOVB #1, waithold

LDX #16

LDY #10

passchecks: CPX #20

BEQ idcheckjmp

PSHX

LDD #disp5

JSR display\_string

PULX

JSR hexkeypad1

CMPA #$FF

BEQ passchecks

LDAB switchchange ; allow for multiple switch changes in the file

CMPB #1

BEQ revertJMP

MOVB #0, wait

CMPA #9

BLE number3s

ADDA #$7

number3s: STAA $411 ;Address 411 contains button pressed for password

LDAB disp2, x

SUBB #$30

CMPB $411 ; check if correct password digit pressed

BNE WRONGjmp

INX

LDAA #'\*' ; indicate on LCD where user is at in password

STAA disp5, y

INY

BRA passchecks

WRONGjmp: JMP WRONG

idcheckjmp: jmp idcheck

revertJMP: MOVB #1, delayon ; same process from the beginning but allows for multiple

; switches to be changed in file

MOVB #2, startuptimer

MOVB #0, delaytimer

switchdelay11:

LDAA delaytimer

CMPA #1

BNE switchdelay11

MOVB #0, switchchange

MOVB #0, delayon

LDAA port\_t

ANDA #$7

STAA switchstatus

LDAA switchstatus

ANDA #1

LDAB prevswitchstatus

ANDB #1

STAB switchchecker

CMPA switchchecker

BEQ nochange11

LDAA ledswitches

ORAA #1

STAA ledswitches

BRA skip011

nochange11:LDAA ledswitches

ORAA #0

BCLR ledswitches, #1

skip011: LDAA switchstatus

ANDA #2

LDAB prevswitchstatus

ANDB #2

STAB switchchecker

CMPA switchchecker

BEQ nochange21

LDAA ledswitches

ORAA #2

STAA ledswitches

BRA skip021

nochange21: LDAA ledswitches

BCLR ledswitches, #2

skip021: LDAA switchstatus

LDAA switchstatus

ANDA #4

LDAB prevswitchstatus

ANDB #4

STAB switchchecker

CMPA switchchecker

BEQ nochange31

LDAA ledswitches

ORAA #4

STAA ledswitches

BRA skip031

nochange31: LDAA ledswitches

BCLR ledswitches, #4

skip031: LDAA switchstatus

CPX #9

BLE passchecki

BRA passchecksjmp

passchecksjmp: jmp passchecks

revertJMPjmp: jmp revertJMP

idcheck: movb #'X',disp5+10

movb #'X',disp5+11

movb #'X',disp5+12 ; display ID length on LCD

movb #'X',disp5+13

LDX #4

LDY #9

passchecki: CPX #8

BEQ updategenerators

PSHX

PSHY

LDD #dispc

JSR display\_string

PULY

PULX

JSR hexkeypad1 ; user inputs ID digit

CMPA #$FF

BEQ passchecki

LDAB switchchange

CMPB #1

BEQ revertJMPjmp ; see if user edited a switch

CMPA #9

BLE number3i

ADDA #$7

number3i: STAA $411 ;Address 411 contains button pressed for password

LDAB disp3, y

SUBB #$30

CMPB $411

BNE WRONG

LDAA #'\*'

STAA dispc, x ; update LCD as user types ID in

INX

INY

BRA passchecki

WRONG: movb #'X',dispc+04

movb #'X',dispc+05 ; if user is wrong, reset LCD id/password string

movb #'X',dispc+06

movb #'X',dispc+07

MOVB #1, holdold

MOVB #0, switchchange

waitforF2: LDD #dispd

JSR display\_string

JSR hexkeypad1 ; wait for user input

MOVB #0, switchchange

CMPA #$F ; verify user understands they made an error

BEQ promptswitchrevertjmp

BRA waitforF2

promptswitchrevertjmp: JMP promptswitchrevert

updategenerators: movb #'X',dispc+04

movb #'X',dispc+05

movb #'X',dispc+06 ; reset LCD id/password string

movb #'X',dispc+07

LDAA gen1au

CMPA #1 ; check if generator is on

BEQ skipturnon1

LDAA port\_t

ANDA #1 ; check status of generator

CMPA #1

BEQ turnon1

MOVB #1, gen1off ; tell program that generator is off

BRA skipturnon1

turnon1: MOVB #0, gen1off ; tell program that generator is on

skipturnon1: LDAA gen2au

CMPA #1

BEQ skipturnon2

LDAA port\_t ; repeat above process for other generators

ANDA #2

CMPA #2

BEQ turnon2

MOVB #1, gen2off

BRA skipturnon2

turnon2: MOVB #0, gen2off

skipturnon2: LDAA gen3au

CMPA #1

BEQ HOMEjmp

LDAA port\_t

ANDA #4

CMPA #4

BEQ turnon3

MOVB #1, gen3off

BRA HOMEjmp

turnon3: MOVB #0, gen3off

HOMEjmp: MOVB #0, switchchange

MOVB #0, holdold

PULD

PULY

PULX

MOVB #0, noTRACKER

MOVB #0, switchflg

LDAA gen1cap

CMPA #0

BNE gen2capc

MOVB #1, gen1off ; let program know generator is off if empty

gen2capc: LDAA gen2cap

CMPA #0

BNE gen3capc

MOVB #1, gen2off ; let program know generator is off if empty

gen3capc: LDAA gen3cap

CMPA #0

BNE exit

MOVB #1, gen3off ; let program know generator is off if empty

exit: RTS

promptswitchrevert: MOVB prevswitchstatus, switchstatus ; a wrong id/password prompts user

; to restore previous switch status

waitforswitch: LDD #dispe

JSR display\_string

MOVB #1, switchflg

LDAA port\_t

LDAB shutoff

CMPB #$1 ; check for shutdown

BEQ shutdownswitch

ANDA #$7

CMPA switchstatus ; check that switches restored

BEQ HOMEjmp

BRA waitforswitch

shutdownswitch: MOVB #0, shutoff

JSR shutdown ; go to shutdown file

BRA waitforswitch

hexkeypad1:

; similar to hexkeypad file except with some modifications that allow switch file to perform

PSHX

PSHY

loop1: LDX #rows

loop11: CPX #rows+4

BEQ loop1

LDAA shutoff

CMPA #1

BEQ shutdown11

PSHD

PSHX

PSHY

JSR read\_pot

LDD pot\_value

STD power\_output

LDAA real\_power\_output

TAB

LDAA #0

PSHX

PSHY

LDY #$0

LDX #$3

IDIV

STX value

LDAA gen1off

CMPA #$1

BEQ skip1

LDAA sum

ADDA real\_value

STAA sum

skip1:

LDAA gen2off

CMPA #$1

BEQ skip2

LDAA sum

ADDA real\_value

STAA sum

skip2:

LDAA gen3off

CMPA #$1

BEQ skip3

LDAA sum

ADDA real\_value

STAA sum

skip3:

LDAA sum

STAA TON

PULY

PULX

PULD

PULX

PULY

MOVB #0, sum

LDAA switchchange

CMPA #1

BNE checktrack1

BRA gohome1

shutdown11: MOVB #0, shutoff

JSR shutdown

BRA gohome1

checktrack1: LDAA TRACKER

CMPA #0

BNE gohome1

LDAA 1, x+

STAA PTU

JSR debounce1

ANDA #$0F

CMPA #$0F

BEQ loop11jmp

BRA letgo1

loop11jmp: JMP loop11

letgo1: LDAA PTU

ANDA #$0F

CMPA #$0F

BNE letgo1

LDAA #0

LDY #lookup

loop21: CMPB 1, y+

BEQ gohome1

inca

CPY #lookup + 16

BNE loop21

BRA loop1JMP

loop1JMP: JMP loop1

gohome1:

PULY

PULX

RTS

debounce1:

JSR delay1

LDAA PTU

TAB

RTS

delay1:

LDY #1000

loop41: DEY

BNE loop41

RTS

auto: MOVB #0, times8 ;AUTO SHUTOFF

gen1b: LDAA #$1 ;store current generator into address

STAA $450

LDAA #'-' ; menu manipulation

STAA disp9

LDAA #'>' ;move arrow to gen 1

STAA disp9+1

LDAA #' '

STAA disp9+17

STAA disp9+16 ; clear arrow from other areas

gen1ab: LDD #disp9

JSR display\_string

JSR hexkeypad1 ; wait for user input

CMPA #$B

BEQ gen2b

CMPA #$A

BEQ gen3b ; if F is pressed select gen 1, if b was pressed shift arrow

CMPA #$F

BEQ JUMPPICKb

BRA gen1ab

JUMPPICKb: JMP GenPickb ; GenPick is too far, needed a jump

gen2b: LDAA #$2 ;stor current generator into address

STAA $450

LDAA #'-'

STAA disp9+16 ; move arrow next to gen 2

LDAA #'>'

STAA disp9+17

LDAA #' '

STAA disp9 ; clear arrow from other areas

STAA disp9+1

gen2ab: LDD #disp9

JSR display\_string

JSR hexkeypad1

CMPA #$B

BEQ gen3b

CMPA #$A ; if b is pressed, move arrow to gen 3, if f is pressed select gen 2, if a is pressed shift arrow to gen 1

BEQ gen1b

CMPA #$F

BEQ JUMPPICKb

BRA gen2ab

gen3b: LDAA #$3 ; store current generator into address

STAA $450

movb #' ',dispa+18

movb #' ',dispa+19

movb #' ',dispa+20

movb #' ',dispa+21

LDAA #'-' ; move arrow to gen3

STAA dispa

LDAA #'>'

STAA dispa+1

LDAA #' ' ; erase arrow from other areas

STAA dispa+16

STAA dispa+17

gen3ab: LDD #dispa

JSR display\_string

JSR hexkeypad1

CMPA #$B

BEQ gen1bJUMP

CMPA #$A ; if b is pressed, move arrow to home, if a is pressed move arrow to gen 2, if f is pressed, select gen 3

BEQ gen2b

CMPA #$F

BEQ GenPickb

BRA gen3ab

gen1bJUMP: JMP gen1b

GenPickb:

LDAA $450

CMPA #$1

BEQ Time1jmp

LDAA $450

CMPA #$2

BEQ Time2jmp

LDAA $450

CMPA #$3

JMP Time3

Time2jmp: JMP Time2

Time1jmp: JMP Time1

off: movb #'G',disp

movb #'e',disp+1

movb #'n',disp+2

movb #'e',disp+3

movb #'r',disp+4

movb #'a',disp+5

movb #'t',disp+6

movb #'o',disp+7

movb #'r',disp+8

movb #' ',disp+9

movb #'i',disp+10

movb #'s',disp+11

movb #' ',disp+18

movb #'o',disp+19

movb #'f',disp+20

movb #'f',disp+21

movb #' ',disp+22

movb #' ',disp+23

movb #' ',disp+24

movb #' ',disp+25

movb #' ',disp+26

movb #' ',disp+27

LDD #disp

JSR display\_string

tryagain: JSR hexkeypad1

CMPA #$F

BNE tryagain

movb #' ',disp

movb #' ',disp+1

movb #' ',disp+2

movb #' ',disp+3

movb #' ',disp+4

movb #' ',disp+5

movb #' ',disp+6

movb #' ',disp+7

movb #' ',disp+8

movb #' ',disp+9

movb #' ',disp+10

movb #' ',disp+11

movb #' ',disp+18

movb #' ',disp+19

movb #' ',disp+20

movb #' ',disp+21

RTS

Time1: LDAA gen1off ; check if generator is even on

CMPA #1

BNE skipo1

JMP off

skipo1: movb #'S',disp+3

movb #'e',disp+4

movb #'t',disp+5

movb #' ',disp+6

movb #'t',disp+7

movb #'i',disp+8

movb #'m',disp+9

movb #'e',disp+10

movb #' ',disp+11

movb #'X',disp+18

movb #'X',disp+19

movb #' ',disp+20

movb #'S',disp+21

movb #'e',disp+22

movb #'c',disp+23

movb #'o',disp+24

movb #'n',disp+25

movb #'d',disp+26

movb #'s',disp+27

LDD #disp

JSR display\_string

dummy1:

JSR hexkeypad1 ; user input

CMPA #$A

BHS dummy1 ; don’t except letter input

ADDA #$30

STAA disp+18 ; display chosen number on LCD

PSHD

LDD #disp

JSR display\_string

PULD

SUBA #$30

LDAB #10

MUL

STAB time1off

dummy1a: ; repeat above process for ones digit of time to turn off

JSR hexkeypad1

CMPA #$A

BHS dummy1a

ADDA #$30

STAA disp+19

PSHD

LDD #disp

JSR display\_string

PULD

SUBA #$30

ADDA time1off

STAA time1off

MOVB #1, autoshut1

LDAA time1off

CMPA #0

movb #'H',dispa+18

movb #'o',dispa+19

movb #'m',dispa+20

movb #'e',dispa+21

movb #' ',disp+18

movb #' ',disp+19

movb #' ',disp+20

movb #' ',disp+21

movb #' ',disp+22

movb #' ',disp+23

movb #' ',disp+24

movb #' ',disp+25

movb #' ',disp+26

movb #' ',disp+27

MOVB #1, g1a

RTS

Time2: LDAA gen2off ; same process but for generator 2

CMPA #1

BNE skipo2

JMP off

skipo2:

movb #'S',disp+3

movb #'e',disp+4

movb #'t',disp+5

movb #' ',disp+6

movb #'t',disp+7

movb #'i',disp+8

movb #'m',disp+9

movb #'e',disp+10

movb #' ',disp+11

movb #'X',disp+18

movb #'X',disp+19

movb #' ',disp+20

movb #'S',disp+21

movb #'e',disp+22

movb #'c',disp+23

movb #'o',disp+24

movb #'n',disp+25

movb #'d',disp+26

movb #'s',disp+27

LDD #disp

JSR display\_string

dummy2: JSR hexkeypad1

CMPA #$A

BHS dummy2

ADDA #$30

STAA disp+18

PSHD

LDD #disp

JSR display\_string

PULD

SUBA #$30

LDAB #10

MUL

STAB time2off

dummy2a: JSR hexkeypad1

CMPA #$A

BHS dummy2a

ADDA #$30

STAA disp+19

PSHD

LDD #disp

JSR display\_string

PULD

SUBA #$30

ADDA time2off

STAA time2off

MOVB #1, autoshut2

MOVB #1, g2a

movb #'H',dispa+18

movb #'o',dispa+19

movb #'m',dispa+20

movb #'e',dispa+21

movb #' ',disp+18

movb #' ',disp+19

movb #' ',disp+20

movb #' ',disp+21

movb #' ',disp+22

movb #' ',disp+23

movb #' ',disp+24

movb #' ',disp+25

movb #' ',disp+26

movb #' ',disp+27

RTS

Time3: LDAA gen3off ; same process for generator 3

CMPA #1

BNE skipo3

JMP off

skipo3:

movb #'S',disp+3

movb #'e',disp+4

movb #'t',disp+5

movb #' ',disp+6

movb #'t',disp+7

movb #'i',disp+8

movb #'m',disp+9

movb #'e',disp+10

movb #' ',disp+11

movb #'X',disp+18

movb #'X',disp+19

movb #' ',disp+20

movb #'S',disp+21

movb #'e',disp+22

movb #'c',disp+23

movb #'o',disp+24

movb #'n',disp+25

movb #'d',disp+26

movb #'s',disp+27

LDD #disp

JSR display\_string

dummy3: JSR hexkeypad1

CMPA #$A

BHS dummy3

ADDA #$30

STAA disp+18

PSHD

LDD #disp

JSR display\_string

PULD

SUBA #$30

LDAB #10

MUL

STAB time3off

dummy3a: JSR hexkeypad1

CMPA #$A

BHS dummy3a

ADDA #$30

STAA disp+19

PSHD

LDD #disp

JSR display\_string

PULD

SUBA #$30

ADDA time3off

STAA time3off

MOVB #1, autoshut3

movb #'H',dispa+18

movb #'o',dispa+19

movb #'m',dispa+20

movb #'e',dispa+21

movb #' ',disp+18

movb #' ',disp+19

movb #' ',disp+20

movb #' ',disp+21

movb #' ',disp+22

movb #' ',disp+23

movb #' ',disp+24

movb #' ',disp+25

movb #' ',disp+26

movb #' ',disp+27

MOVB #1, g3a

RTS