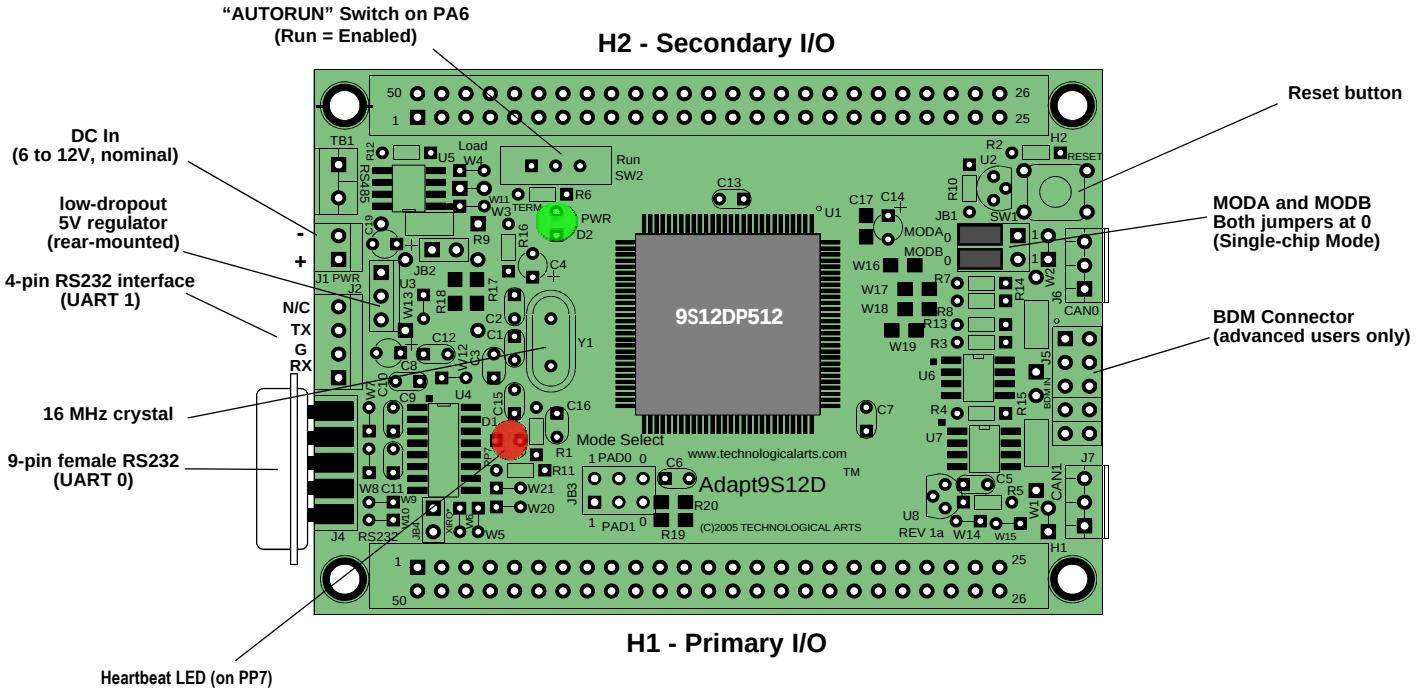
Adapt9S12DP512 Module with on-chip BASIC



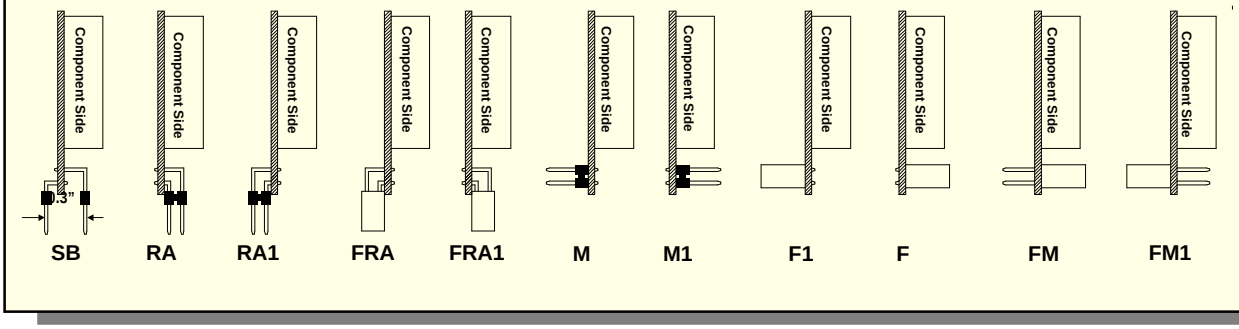
CONNECTOR PIN ASSIGNMENTS

H1				H2				
PIN	SIGNAL NAME	PIN	SIGNAL NAME	PIN	SIGNAL NAME	PIN	SIGNAL NAME	
1	PS4	50	GROUND	1	PA7	50	VCC (+5VDC)	
2	PS5	49	GROUND	2	PA6 (Autorun Switch)	49	GROUND	
3	PS6	48	PS0 (UART0 RXD)	3	PA5	48	PE7	
4	PS7	47	+5VDC	4	PA4	47	PK7	
5	PS1 (UART0 TXD)	46	PE1 (ZigFlea irq*)	5	PA3	46	PK5	
6	PT7	45	PE0/XIRQ*	6	PA2	45	PK4	
7	PT6	44	RESET*	7	PA1	44	PK3	
8	PT5	43	PE7	8	PA0	43	PK2	
9	PT4	42	PH0	9	PB7	42	PK1	
10	PT3	41	PH1	10	PB6	41	PK0	
11	PT2	40	PH2	11	PB5	40	PJ0	
12	PT1	39	PH3	12	PB4	39	PJ7 (SCL)	I2C pins
13	PT0	38	PH4	13	PB3	38	PJ6 (SDA)	
14	PP7 (Heartbeat LED)	37	PH5	14	PB2	37	PM7	
15	PP6	36	PH6	15	PB1	36	PM6	
16	PP5	35	PH7	16	PB0	35	PM5/SCK	
17	PP4	34	PS2/RXD1	17	PE2	34	PM4/MOSI	QSPI pins
18	PP3	33	PE4 (ZigFlea attn*)	18	PE4 (ZigFlea attn*)	33	PM3/SS*	
19	PP2	32	PS3/TXD1	19	PE3 (ZigFlea rxten)	32	PM2/MISO	
20	PP1	31	VRL	20	PE1 (ZigFlea irq*)	31	PM1	
21	PP0	30	VRH	21	PJ1 (ZigFlea rst*)	30	PM0	
22	PAD00	29	PAD04	22	PAD08	29	PAD12	
23	PAD01	28	PAD05	23	PAD09	28	PAD13	
24	PAD02	27	PAD06	24	PAD10	27	PAD14	
25	PAD03	26	PAD07	25	PAD11	26	PAD15	

ZigFlea pins[1]

NOTES: * indicates active low signal

Standard Connector Options (use NC for "no connector")



1 - Recommended pins for implementing ZigFlea feature. Requires addition of external hardware.

Order Code: Module with StickOS BASIC in Flash: AD9S12DP512BM0-❑-❑

(NOTE: When ordering modules, fill in -❑-❑ with desired connector option codes for H1 and H2, as shown above)

www.technologicalarts.com • sales@technologicalarts.com • phone: +1(416) 963-8996 • fax: +1(416) 963-9179

StickOS BASIC Language Quick Reference (v1.90) - for 9S12DP512 -

Commands

<Ctrl-C> stop running program
auto [*line*] automatically number program lines
clear [**flash**] clear ram [and flash] variables
cls clear terminal screen
cont [*line*] continue program from stop
delete ([*line*][-[*line*]]|*subname*) delete program lines
dir list saved programs
edit *line* edit program line
help [*topic*] online help
list ([*line*][-[*line*]]|*subname*) list program lines
load *name* load saved program
memory print memory usage
new erase code ram and flash memories
profile ([*line*][-[*line*]]|*subname*) display profile info
purge *name* purge saved program
renumber [*line*] renumber program lines (and save)
reset reset the MCU!
run [*line*] run program
save [*name*|**library****] save code ram to flash memory
upgrade upgrade StickOS firmware!
uptime print time since last reset

Modes

analog [*millivolts*] set analog voltage scale
autorun [**on**|**off**] autorun mode (on reset)
baud [*rate*] UART transport baud rate (on reset)
echo [**on**|**off**] terminal echo mode
indent [**on**|**off**] listing indent mode
keychars [*keychars*] set/display keypad scan chars
numbers [**on**|**off**] listing line numbers mode
pins [*assign* [*pinname*|**none**]] set/display pin assignments
prompt [**on**|**off**] terminal prompt mode
servo [*Hz*] set/display servo Hz (on reset)
step [**on**|**off**] debugger single-step mode
trace [**on**|**off**] debugger trace mode
watchsmart [**on**|**off**] low-overhead watchpoint mode

General Statements

line delete program line
*line statement // comment** enter program line
variable[**$**] **=** *expression*, ... ** assign variable
? [**dec**|**hex**|**raw**] *expression*, ...[;] ** print strings/expressions
assert *expression* break if expression is false
data *n* [, ...] read-only data
dim *variable*[**$**][[*n*]] [as ...], ... dimension variables
end end program
halt loop forever
input [**dec**|**hex**|**raw**] *variable*[**$**], ... input data
label *label* read/data label
lcd *pos*, [**dec**|**hex**|**raw**] *expression*, ... * display results on lcd
let *variable*[**$**] **=** *expression*, ... assign variable
print [**dec**|**hex**|**raw**] *expression*, ...[;] print strings/expressions
read *variable* [, ...] read data into variables
rem *remark* remark
restore [*label*] restore data pointer
sleep *expression* (**s**|**ms**|**us**) delay program execution
stop insert breakpoint in code
vprint *var*[**$**] **=** [**dec**|**hex**|**raw**] *expr*, ... print to variable

Block Statements

if *expression* **then**
[**elseif** *expression* **then**]
[**else**]
endif
for *variable* **=** *expression* **to** *expression* [**step** *expression*]
[(**break**|**continue**) [*n*]]
next
while *expression* **do**
[(**break**|**continue**) [*n*]]
endwhile
do
[(**break**|**continue**) [*n*]]
until *expression*
gosub *subname* [*expression*, ...]
sub *subname* [*param*, ...]
[**return**]
endsub

Device Statements

timers:
configure timer *n* **for** *n* (**s**|**ms**|**us**)
on timer *n* **do** *statement*
off timer *n* disable timer interrupt
mask timer *n* mask/hold timer interrupt
unmask timer *n* unmask timer interrupt
uarts:
configure uart *n* **for** *n* **baud** *n* **data** \ (**even**|**odd**|**no**) **parity** [**loopback**]
on uart *n* (**input**|**output**) **do** *statement*
off uart *n* (**input**|**output**) disable uart interrupt
mask uart *n* (**input**|**output**) mask/hold uart interrupt
unmask uart *n* (**input**|**output**) unmask uart interrupt
uart *n* (**read**|**write**) *variable*, ... perform uart I/O
i2c:
i2c start *addr* master i2c I/O
i2c (**read**|**write**) *variable*, ...
i2c stop
qspi:
qspi *variable* **[, ...]** master qspi I/O
watchpoints:
on *expression* **do** *statement*
off *expression* disable expr watchpoint
mask *expression* mask/hold expr watchpoint
unmask *expression* unmask expr watchpoint

Expressions

the following operators are supported as in C, in order of decreasing precedence:
n decimal constant
0x*n* hexadecimal constant
'*c***'** character constant
variable simple variable
*variable***[***expression***]** array variable element
*variable***#** length of array or string
( ) grouping
! ~ logical not, bitwise not
*** / %** times, divide, mod
+ - plus, minus
>> << shift right, left
<= < >= > inequalities
== != equal, not equal
| ^ & bitwise or, xor, and
|| ^^ && logical or, xor, and

Strings

v$ is a nul-terminated view into a byte array v[]
string statements:
dim, **input**, **let**, **print**, **vprint**
if *expression relation expression* **then**
while *expression relation expression* **do**
until *expression relation expression*
string expressions:
"*literal***"** literal string
*variable***$** variable string
*variable***$[***start***:***length***]** variable substring
+ concatenates strings
string relations:
<= < >= > inequalities
== != equal, not equal
~ !~ contains, does not contain

Variables

all variables must be dimensioned!
variables dimensioned in a sub are local to that sub
simple variables are passed to sub params by reference
array variable indices start at 0
v is the same as v[0], except for input/print/i2c/qspi statements
ram variables:
dim *var*[**$**][[*n*]]
dim *var*[[*n*]] **as** (**byte**|**short**)
flash parameter variables:
dim *varflash*[[*n*]] **as flash**
pin alias variables:
dim *varpin*[[*n*]] **as pin** *pinname* **for** \
(**digital**|**analog**|**servo**|**frequency**|**uart**) \ (**input**|**output**) \
[**debounced**] [**inverted**] [**open_drain**]
absolute variables:
dim *varabs*[[*n*]] as (byte|short)**at address** *addr*
dim *varabs*[[*n*]] **as** (**byte**|**short**) **at address** *addr*
system variables (read-only):
analog* **getchar keychar*** **msecs random**** **seconds ticks ticks_per_msec**

Pins

Use the "**help pins**" command to see MCU-specific pin names and capabilities; use the "**pins**" command to set/display pin assignments
* = v1.82 and later; ** = v1.90 and later
Note that as of v1.84, the units of servo output pins was changed from centi-milliseconds (cms) to microseconds (us)

Pins

pin names:

0	1	2	3	4	5	6	7	
pad00	pad01	pad02	pad03	pad04	pad05	pad06	pad07	PORT AD0
pad08	pad09	pad10	pad11	pad12	pad13	pad14	pad15	PORT AD1
pa0	pa1	pa2	pa3	pa4	pa5	pa6	pa7	PORT A
pb0	pb1	pb2	pb3	pb4	pb5	pb6	pb7	PORT B
pe0	pe1	pe2	pe3	pe4	pe5	pe6	pe7	PORT E
ph0	ph1	ph2	ph3	ph4	ph5	ph6	ph7	PORT H
pj0	pj1					pj6	pj7	PORT J
pk0	pk1	pk2	pk3	pk4	pk5	pk6	pk7	PORT K
pm0	pm1	pm2	pm3	pm4	pm5	pm6	pm7	PORT M
pp0	pp1	pp2	pp3	pp4	pp5	pp6	pp7	PORT P
ps0	ps1	ps2	ps3	ps4	ps5	ps6	ps7	PORT S
pt0	pt1	pt2	pt3	pt4	pt5	pt6	pt7	PORT T

Notes on Pin Names:

Some pins have been factory-assigned to suit the board configuration

All pins support general purpose digital input
All pins except pad?? and pe[01] support general purpose digital output
pad?? = potential analog input pins (mV)
pj6, pj7 support I2C (two-wire interface)
pp? = potential analog output (PWM) pins (mV)
pp? = potential servo output (PWM) pins (us)
pt? = potential frequency output pins (Hz)
ps0 (u0), ps2 (u1) = potential uart input pins (received byte)
ps1 (u0), ps3 (u1) = potential uart output pins (transmit byte)

Visit www.soBASICsoEasy.com for complete details