

PUB start: wflag n	'system: initialisiert system
PUB startram	'system: initialisierung des systems bei ram-upload
PUB paraset(stradr) i,c	'system: parameterbereich beschreiben
PUB parastart	'system: setzt den zeiger auf parameteranfangsposition
PUB paradel i	'system: parameterbereich löschen
PUB paranext(stradr): err i,c	'system: überträgt den nächsten parameter in stringdatei
PUB blood(stradr) n,rc,ii,plen	'system: bellatrix mit grafiktreiber initialisieren
PUB breset	'system: bellatrix neu starten
PUB stop	'loader: beendet anwendung und startet os
PUB ldsys	'loader: startet sys.bin
PUB ldbin(stradr) len,i,stradr1,stradr2	'loader: startet bin-datei über loader
PUB sdmount err	'sd-card: mounten
PUB sdclose:err	'sd-card: datei schließen
PUB sdopen(modus,stradr):err len,i	'sd-card: datei öffnen
PUB sdgetc: char	'sd-card: zeichen aus datei lesen
PUB sdputc(char)	'sd-card: zeichen in datei schreiben
PUB sddir	'sd-card: verzeichnis wird geöffnet
PUB sdnext: stradr flag,len,i	'sd-card: nächster dateiname aus verzeichnis
PUB sdeof: fl_eof	'sd-card: eof-flag abfragen
PUB sdseek(wert)	'sd-card: zeiger auf bytewidth setzen
PUB sdfattrib: attrib wert	'sd-card: dateiattribute abfragen
PUB sdftime: ftime wert	'sd-card: zeitstempel abfragen
PUB sdfsize: fsize wert	'sd-card: dateigröße abfragen
PUB si_open(stradr): err len,i	'scr: containerdatei öffnen
PUB si_fill(char)	'scr: screenpuffer mit zeichen füllen
PUB si_read(snr)	'scr: screen in den puffer laden
PUB si_write(snr)	'scr: screen auf disk schreiben
PUB si_getnr: snr	'scr: nummer des aktuellen screens abfragen
PUB si_setpos(pos)	'scr: zeiger auf position im puffer setzen
PUB si_getpos: pos	'scr: aktuelle position im puffer abfragen
PUB si_getc: char	'scr: zeichen wird aus dem puffer gelesen
PUB si_putc(char)	'scr: zeichen wird in den puffer geschrieben
PUB si_flush	'scr: aktuellen puffer auf disk schreiben
PUB si_err: err	'scr: fehlerstatus abfragen
PUB si_maxscr: snr	'scr: anzahl der screens des containers abfragen
PUB hss_playfile(stradr) status	'hss: spielt übergebene hss-datei von sd-card
PUB hss_stop	'hss: stoppt aktuellen song
PUB hss_pause	'hss: pausiert aktuellen song
PUB hss_load(stradr): status len,i	'hss: lädt hss-datei von sd-card in songpuffer
PUB hss_play	'hss: spielt song im puffer ab
PUB hss_vol(vol)	'hss: volume einstellen 0..15
PUB hss_peek(n): wert	'hss: registerwert auslesen
PUB hss_intreg(n): wert	'hss: interfacerregister auslesen
PUB sfx_setslot(adrs,slot) i,n	'sfx: sendet sfx-daten in sfx-slot
PUB sfx_fire(slot,chan)	'sfx: triggert einen bestimmten soundeffekt
PUB wav_play(stradr): status len,i	'sdw: spielt wav-datei direkt von sd-card
PUB wav_stop	
PUB wav_status: status	'sdw: status des players abfragen
PUB ram_read(adresse):wert	'eram: liest ein byte vom eram
PUB ram_write(wert,adresse)	'eram: schreibt ein byte in eram
PUB rd_long(eadr): wert	'eram: liest long ab eadr
PUB rd_word(eadr): wert	'eram: liest word ab eadr

PUB wr_long(wert,eadr) n	'eram: schreibt long ab eadr
PUB wr_word(wert,eadr) n	'eram: schreibt word ab eadr
PUB key:wert	'key: holt tastaturcode
PUB keyspec:wert	'key: statustasten zum letzten tastencode
PUB keystat:status	'key: übergibt tastaturstatus
PUB keywait:n	'key: wartet bis taste gedrückt wird
PUB input(stradr,anz) curpos,i,n	'key: stringeingabe
PUB print(stringptr) terminiert)	'screen: bildschirmausgabe einer zeichenkette (0-
PUB printestr(eadr) i,len (mit längenbyte)	'screen: bildschirmausgabe einer zeichenkette im eram!
PUB printdec(value) i	'screen: dezimalen zahlenwert auf bildschirm ausgeben
PUB printhex(value, digits) ausgeben	'screen: hexadezimalen zahlenwert auf bildschirm
PUB printchar(c)	'screen: einzelnes zeichen auf bildschirm ausgeben
PUB printctrl(c) ausgeben	'screen: steuerzeichen (\$100 bis \$1FF) auf bildschirm
PUB printnl	'screen: \$0D - CR ausgeben
PUB printcls	'screen: screen löschen
PUB curhome	'screen: cursorposition auf erste position setzen
PUB printtab	'screen: zur nächsten tabulatorposition
PUB curchar(char)	'screen: setzt cursorzeichen
PUB curpos1	'screen: setzt cursor auf spalte 1 in zeile
PUB cursetx(x)	'screen: setzt cursorposition auf x
PUB cursety(y)	'screen: setzt cursorposition auf y
PUB curgetx: x	'screen: abfrage x-position cursor
PUB curgety: y	'screen: abfrage y-position cursor
PUB setcolor(color)	'screen: farbe setzen
PUB curon	'screen: schaltet cursor an
PUB curoff	'screen: schaltet cursor aus
PUB sline(n)	'screen: startzeile scrollbereich setzen
PUB scrollup	'screen: scrollt screen eine zeile hoch
PUB scrolldown	'screen: scrollt screen eine zeile runter
PUB screeninit(stradr,n)	'screen: löschen, kopfzeile ausgeben und setzen
PUB bus_init	'bus: initialisiert bussystem
CON	
PUB bus_putchar1(c)	'bus: byte an prop2 (administra) senden
PUB bus_getchar1:wert	'bus: byte vom prop2 (administra) empfangen
PUB bus_putchar2(c)	'bus: byte an prop1 (bellatrix) senden
PUB bus_getchar2:wert	'bus: byte vom prop1 (bellatrix) empfangen