ＳＦＬ仕様書

第 １ 版　改３

制定　：　パルテノン研究会

第１版　　：　平成１２年 ５月１９日

第１版　改１：　平成１４年 ５月１０日

第１版　改２：　平成２０年 ６月１０日

第１版　改３：　平成２６年 １月　７日

版管理

ＳＦＬ仕様書の改変履歴を以下に示す。

平成２６年1月7日　　第１版改３

・２６ページ「2進数」「8進数」「16進数」の構文図を変更
第１版改２までの構文図および変更理由は、別紙１（２９ページ）を参照

平成２０年6月１０日　第１版改２

・１０ページ「制御端子による動作の定義」の構文図で「制御出力端子名」が「制御入力端子名」となっていたことを訂正

平成１４年5月１０日　第１版改１

・５ページ「被利用時の制御端子の仮引数の定義」の構文図に ; が欠落していることを訂正

・１０ページ「制御端子による動作の定義」の構文図で「制御内部端子名」が「制御入力端子名」となっていたことを訂正

平成１２年5月１９日　第１版制定

ＳＦＬ

ＳＦＬは、単相同期回路を対象とした論理回路記述言語である。

構文図の説明

◯ は、中に記述している文字そのものを示す。

(　　) は、中に記述している文字列そのもの（キーワード）を示す。

[　　] は、他で定義されているものを示す。

⟶ は、構文の流れを示している。

構文図に従った記述を行うには、左端から矢印に従って進み、途中出会った◯ または (　　) 内の文字または文字列を連結していく。

ただし、キーワードまたは [名前] を連結する場合は、間に区切り文字（注参照）を挟む。

キーワードは、[名前] としては使用出来ない。

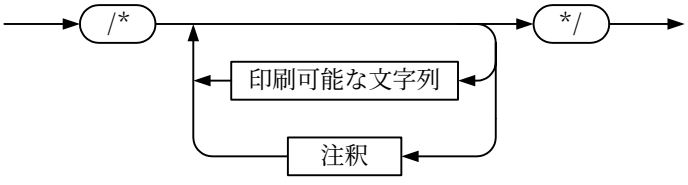
また、[予約語] は、キーワードの扱いに準ずる。

矢印で示す経路に分岐があるときは、任意の経路を選択して進むことが出来る。

入り口から出口まで矢印の示す方向に従って記述を並べていくとＳＦＬ記述が出来上がるが、動作記述の中で使用する名称は、全て事前に定義しておく必要がある。

（注）区切り文字としては、空白文字・タブ文字・改行文字が使用できる。

注釈

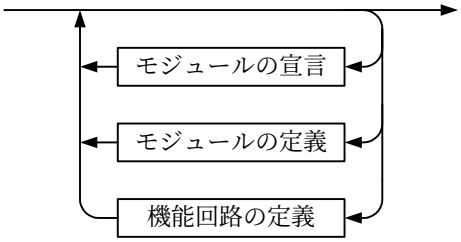


注釈は、単なる説明のための記述であるが、ＳＦＬ記述としては空白と同等に扱われる。

[印刷可能な文字列] は、印刷可能な文字で構成されていれば何でも構わない。

ただし、文字列 (/*) や (*/) は含まないこと。（注釈の多重は構わない。）

ＳＦＬ記述

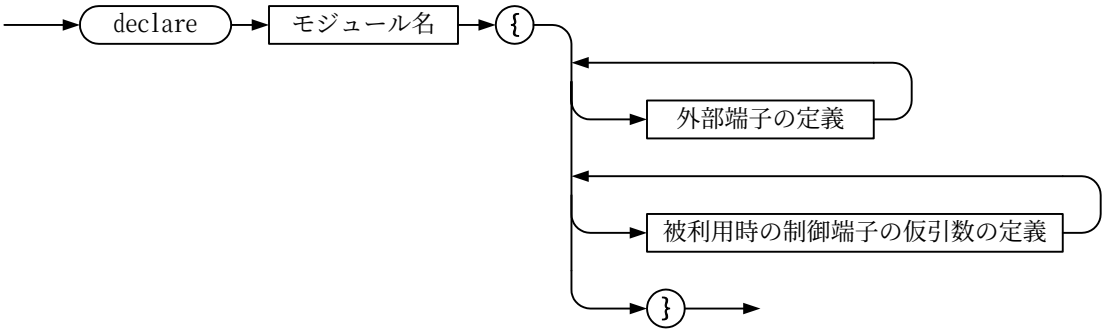


ＳＦＬ記述は、モジュールを基本単位とする階層構造の形式をとる。モジュールを記述する際には、別途記述された他のモジュールを自己の下位部品（サブモジュール）として使用することが許される。

モジュールの定義は、モジュールの定義 によって行う。ここで定義したモジュールを他のモジュールの下位部品として使用する場合には、このモジュールのプロトタイプ宣言である モジュールの宣言 も行う必要がある。

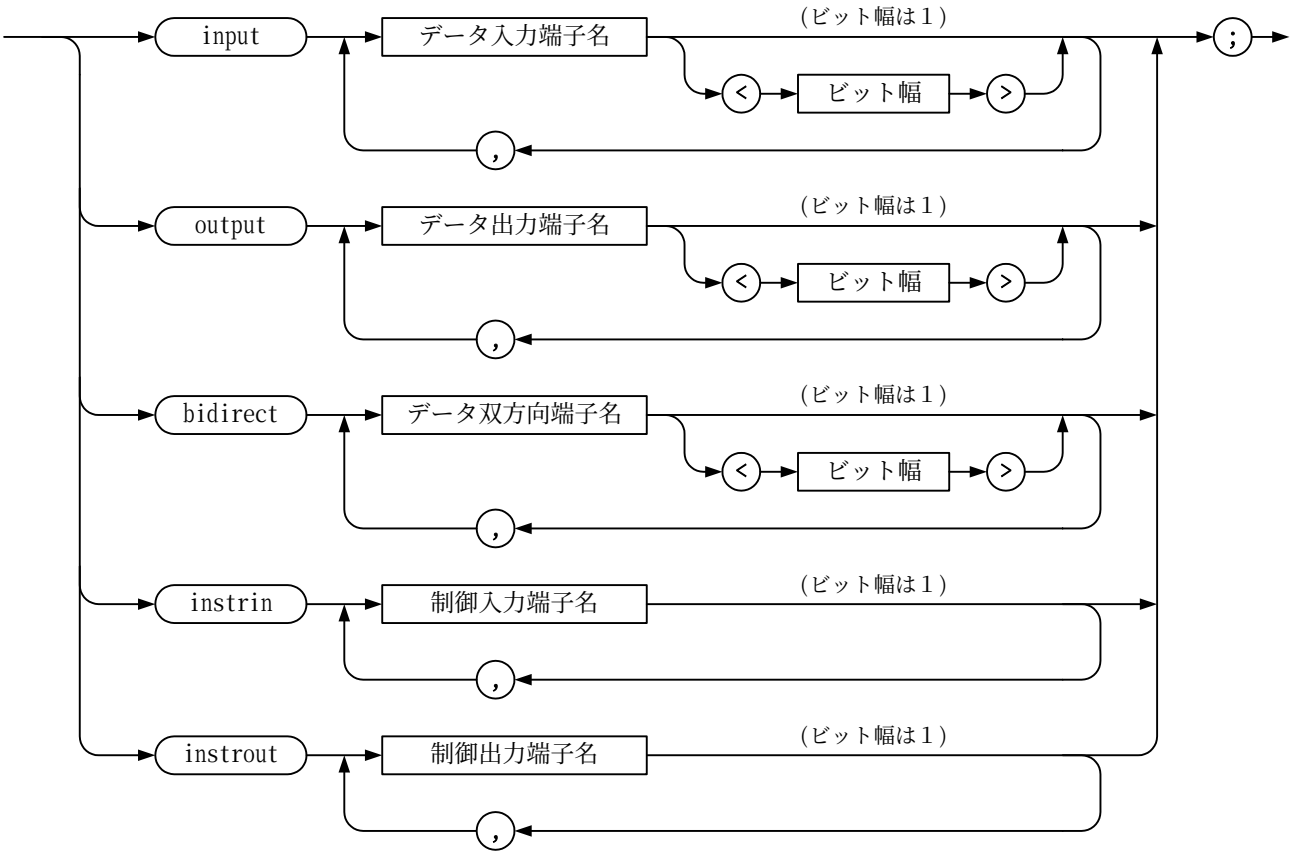
モジュールの定義 は、現実の回路として展開・合成出来るように、使用できる演算子に制限（⊕などは禁止）が加えられている。展開・合成の必要がない（シミュレーションのみの）場合には、全ての演算子が使用可能な 機能回路の定義 でモジュールを定義することができる。

モジュールの宣言



モジュールの定義 で定義したモジュールを他のモジュールの下位部品として使用する前に、この宣言が必要である。この宣言により、モジュールを下位部品として使用する際の外部端子や引数を使用する側に通知する。

外部端子の定義

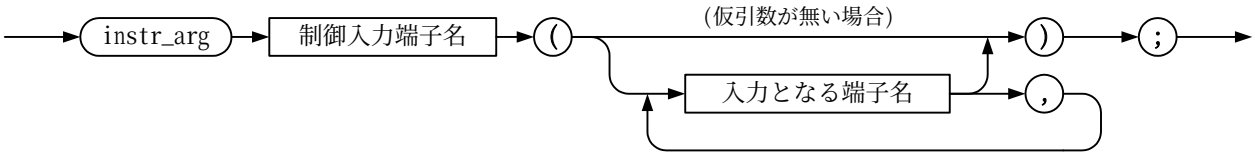


モジュール内の回路と外側の回路を繋ぐための入出力端子を定義する。入出力端子には、単にデータの受け渡しを担うデータ端子と、何らかの制御を伴う制御端子とがある。

データ端子への値の設定は、演算子(=) による値の出力で行う。データ端子が保持する値は、値の出力が指示されたクロック期間内のみ有効であり、値の出力が指示されなかったクロック期間内での値は不定となる。データ端子はビット幅を指定することができる。指定しなかった場合のビット幅は１となる。データ端子に出力する値は、データ端子のビット幅と同一である必要がある。

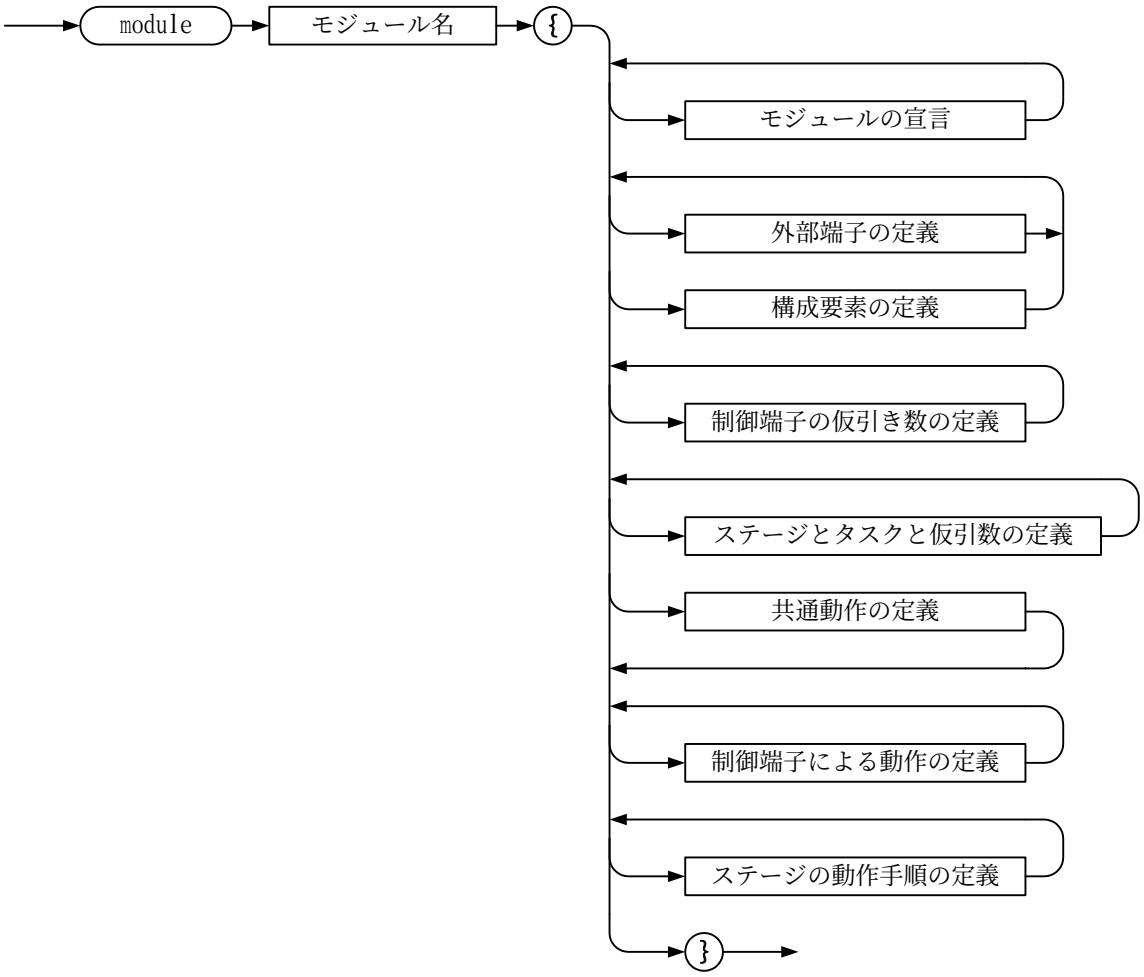
制御端子への値の設定は、単位動作 制御端子の起動 による。制御端子の値は、起動が指示されたクロック期間内では１（ビット幅は１）となり、起動が指示されなかったクロック期間内では０（ビット幅は１）となる。

被利用時の制御端子の仮引数の定義



部品（サブモジュール）として使用するモジュールの制御入力端子を起動する際に、同時に値を設定したい入力端子があれば、仮引数として指定することができる。

モジュールの定義



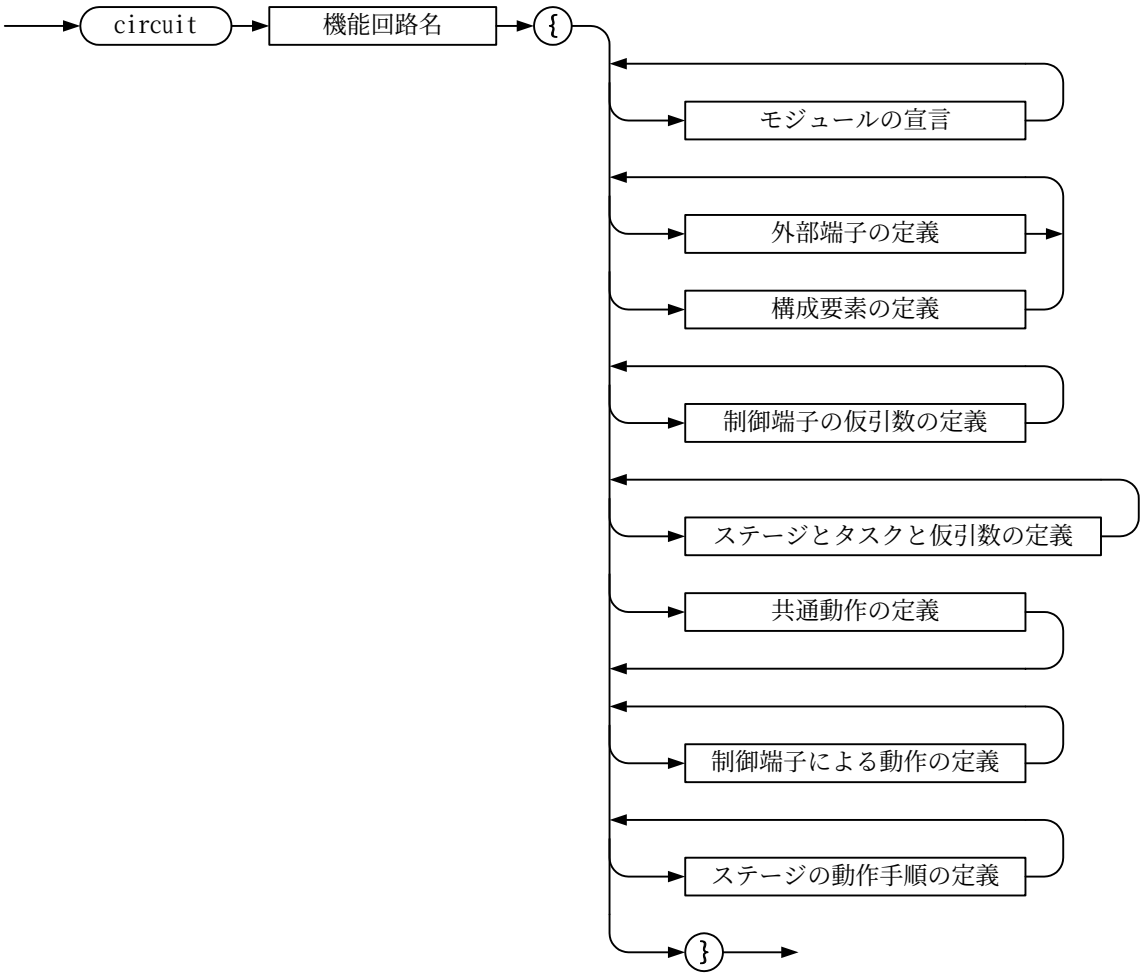
回路をＳＦＬ記述する際の単位であるモジュールを定義する。このモジュールと外部回路とのデータの授受は、このモジュール内の 外部端子の定義 で指定した外部端子を通じて行う。外部回路からモジュールの内部回路を直接操作することは出来ない。

このモジュールが他のモジュールの下位部品（サブモジュール名）となっている場合は、

サブモジュール名．外部端子名

とする。

機能回路の定義



機能回路の定義は、モジュールの定義とほぼ同等であるが、現実の回路の合成を目的としていないため全ての演算子とメモリを自由に使用することができる。機能回路の中だけで使用できる演算子を以下に示す。

(加算)

(ビット右シフト)

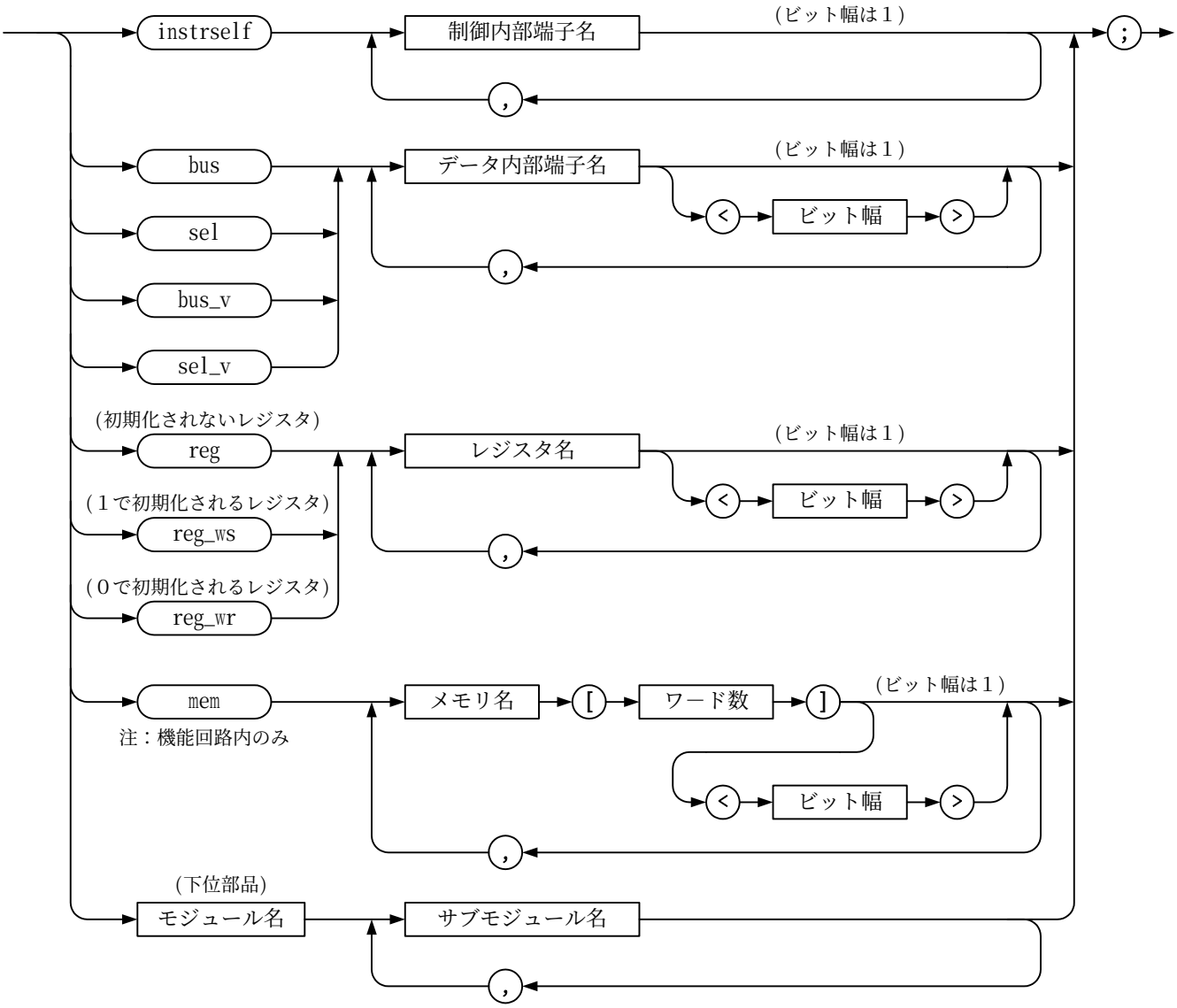
(ビット左シフト)

(一致判定)　右辺の式が定数でないとき

(デコード)

(エンコード)

構成要素の定義

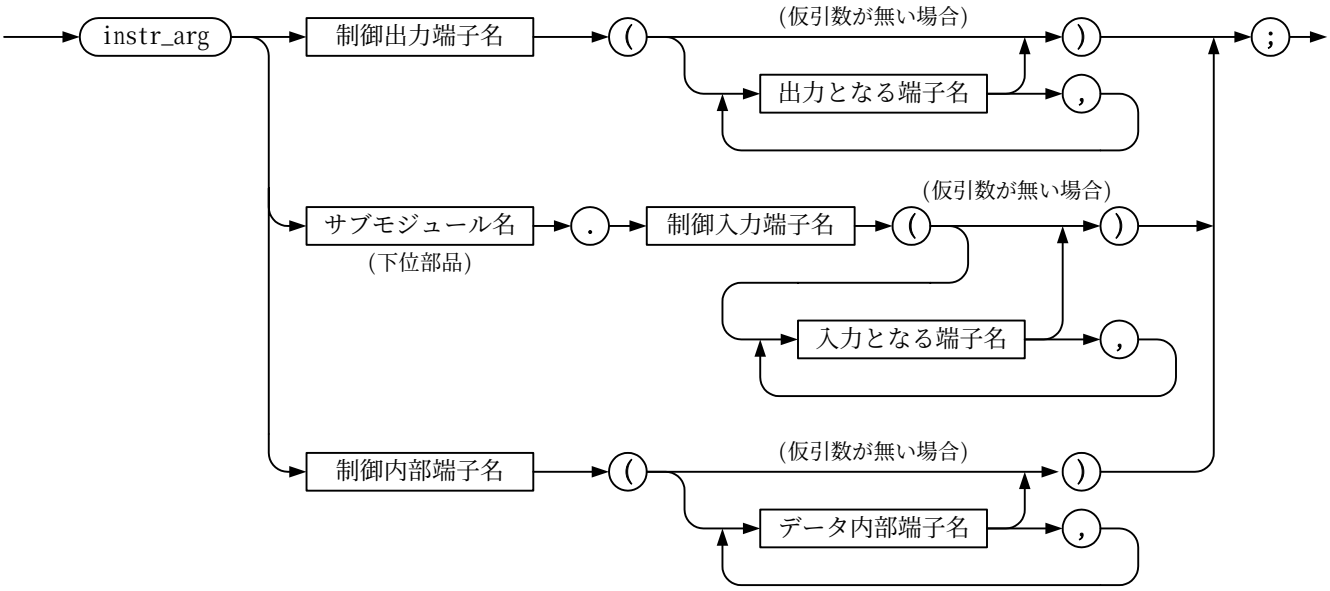


モジュール（および機能回路）内で使用する端子・レジスタ・メモリ・下位部品（サブモジュール）を定義する。

ここで定義しているものは、モジュールの外部から直接参照・操作することは出来ない。

モジュール名 は、下位部品として使用したいモジュールの名前を指定する。この下位部品は、このモジュール内では サブモジュール名 で指定した名前で参照する。

制御端子の仮引数の定義



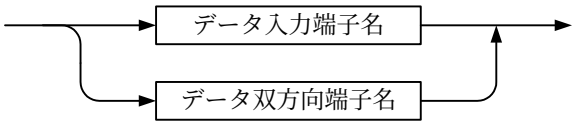
制御端子を起動する際に、同時にデータ端子に値を設定したいデータ端子があれば、それらのデータ端子を起動の仮引数として指定することが出来る。

下位部品（サブモジュール）の制御端子を指定するには、

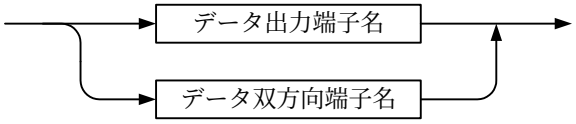
サブモジュール名．外部端子名

とするが、仮引数として指定するデータ端子に「サブモジュール名．」を前置する必要はない。

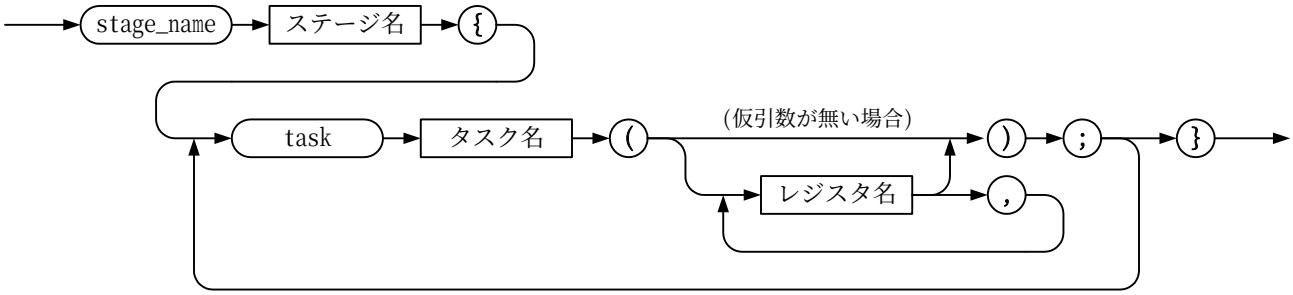
入力となる端子名



出力となる端子名

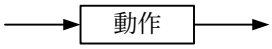


ステージとタスクと仮引数の定義



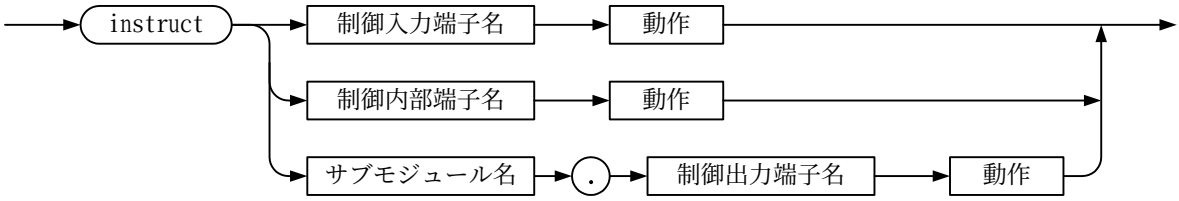
ＳＦＬでは、ステートマシンをステージと呼び、ステージが動作中にとる状態をステート呼ぶ。
ステージは複数のステートをとりうるが、ある１クロックの期間内はそのうちの１つのステートをとり、他のステートを同時にとることはない。ステージがステートを変更する（遷移する）のはクロックの変わり目である。
ステージを始動するに際し、タスクと呼ぶ動作モードを指定する必要がある。１つのステージは複数のタスクを持つことが出来るが、始動に際し指定出来るタスクは１個のみである。また、一旦始動したステージは、その動作を終了するまでは別のタスクを指定して再始動させることは許されない。
タスクはレジスタを仮引数として指定することができ、ステージを始動する際のパラメータをレジスタに設定できる。
タスクは１ビットの値を保持しており、ステージが動作していないとき全てのタスクの値は０であり、ステージが動作しているときは始動にあたり指定したタスクのみが１となる。

共通動作の定義



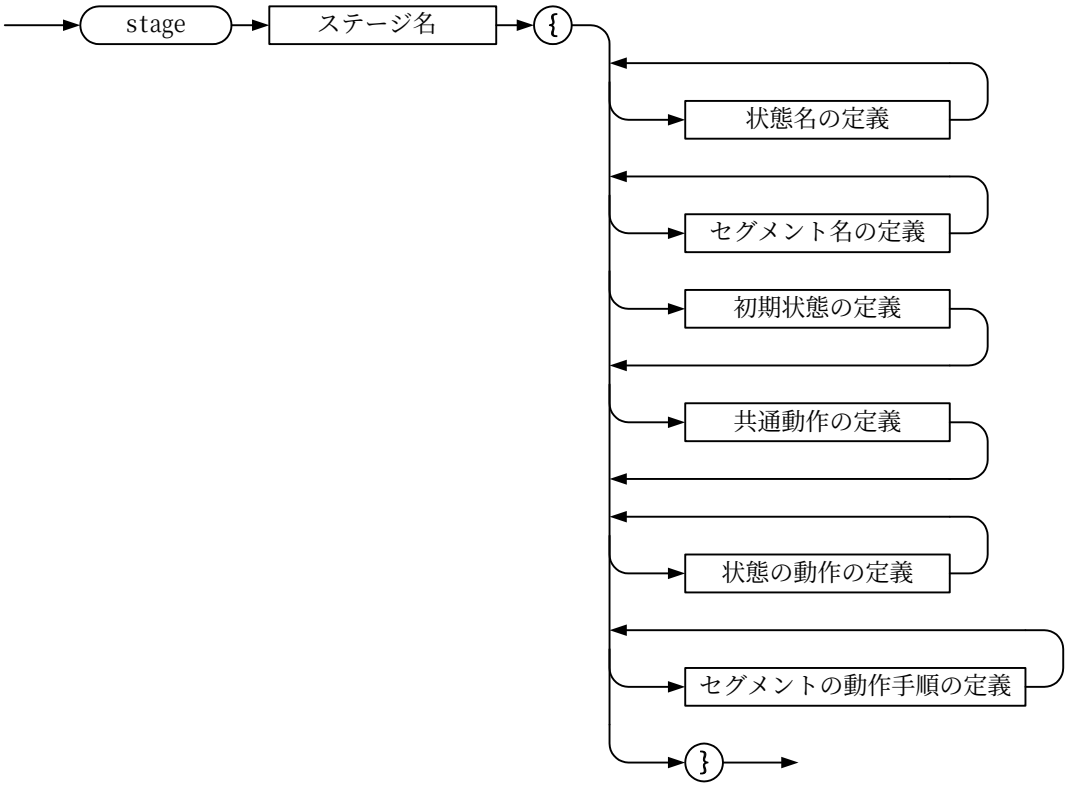
動作 で指定した動作を実行するに際して特定の条件が付かないものは、共通動作として定義する。ただし、ステージまたはセグメントの中で記述された共通動作については、その動作を含むステージまたはセグメントが動作していることが必要条件である。

制御端子による動作の定義



指定した制御端子の値が１の場合に実行する動作を定義する。

ステージの動作手順の定義



ステージの動作を定義する。

ステージがとりうる状態の名前を 状態名の定義 および セグメント名の定義 で列挙する。

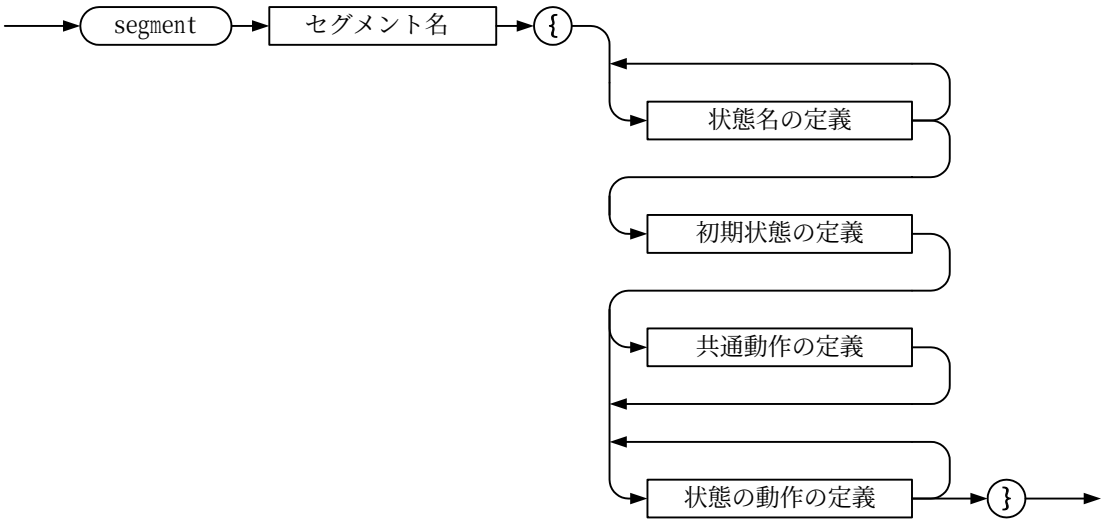
セグメントは、複数のステートを１つの纏まりとして定義したものである。

初期状態の定義 は、回路のリセット後ステージが最初にとる状態（ステート）を指定する。

共通動作の定義 は、このステージが動いている間ずっと実行される動作を定義する。

状態の動作の定義 は、このステージがその状態（ステート）をとった場合に実行される動作を定義する。

セグメントの動作手順の定義



セグメントを定義する。

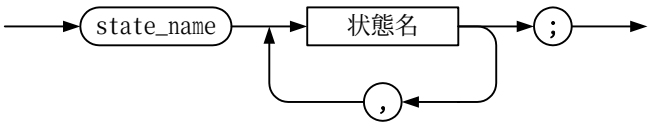
セグメントがとりうる状態の名前を 状態名の定義 で列挙する。

初期状態の定義 は、セグメントが始動したとき最初にとる状態を指定する。

共通動作の定義 は、このセグメントが動いている間ずっと実行される動作を定義する。

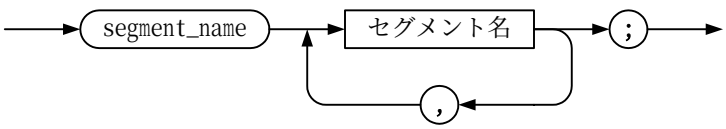
状態の動作の定義 は、このセグメントがそのステート状態をとった場合に実行される動作を定義する。

状態名の定義



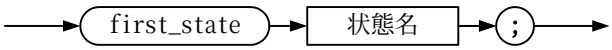
ステージまたはセグメントがとりうる状態の名前を定義する。

セグメント名の定義



ステージが持つセグメントの名前を定義する。

初期状態の定義

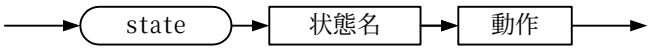


ステージまたはセグメントの初期状態を指定する。

ステージの場合は、リセット後の最初の状態を指定する。

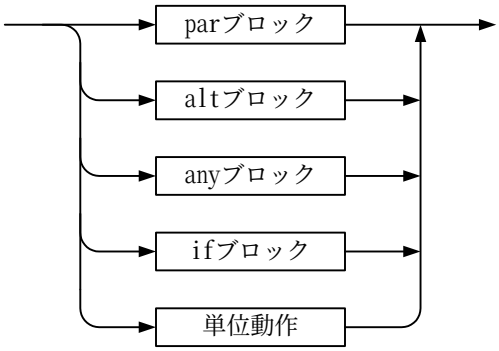
セグメントの場合は、呼び出されたときの最初の状態を指定する。

状態の動作の定義



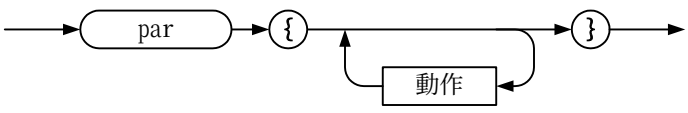
ステージまたはセグメントが 状態名 で指定する状態（ステート）をとった場合に実行する動作を定義する。

動作



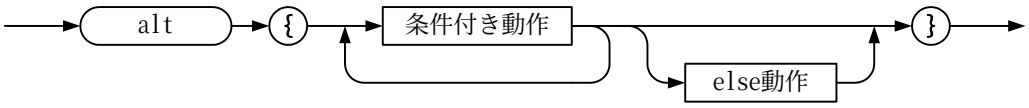
目的の回路を、その動作を記述することによって定義する。

parブロック



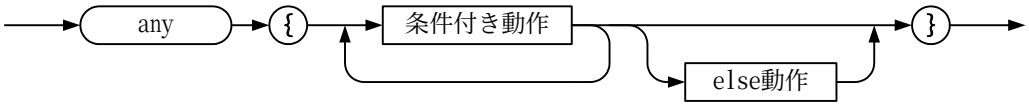
キーワード par は、後に続く複数の 動作 の実行順序が全く並列の関係にあることを示す。

altブロック



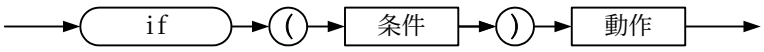
キーワード alt は、後に続く複数の 条件付き動作 の条件評価の順番が出現順であることを示している。この条件評価の過程で条件が成立する 条件付き動作 が見つかれば、後に続く 条件付き動作 および else動作 は、全く無視される。
条件を満たしている 条件付き動作 が見つからない場合のみ else動作 が実行される。

anyブロック



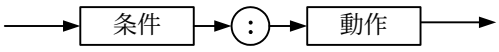
キーワード any は、後に続く複数の 条件付き動作 の条件評価の順番が全く並列であることを示している。この条件評価の過程で条件が成立する 条件付き動作 は、他の結果に関わらず実行される。
条件を満たしている 条件付き動作 が見つからない場合のみ else動作 が実行される。

ifブロック



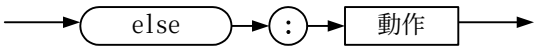
キーワード if は、後に続く 動作 の実行に関して条件が付いていることを示している。
条件 で示す条件が成立した場合のみ 動作 で示す動作が実行され、そうでない場合、
動作 は全く無視される。

条件付き動作



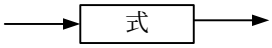
条件 で示す条件が成立した場合のみ 動作 で示す動作が実行され、そうでない場合、
動作 は全く無視される。

else動作



ａｌｔブロックおよびａｎｙブロック内に於いて、条件が成立する 条件付き動作 が全く無かった場合に実行する動作を 動作 で定義する。

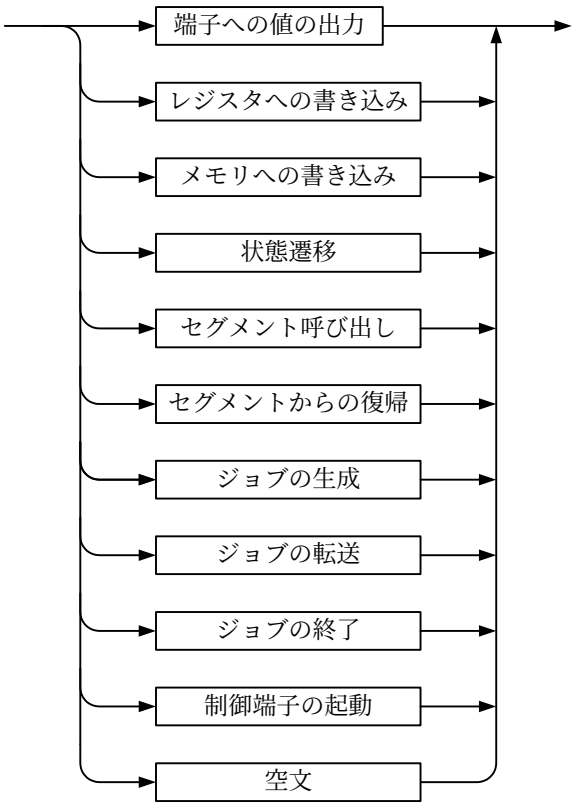
条件



ａｌｔブロック，ａｎｙブロックおよびｉｆブロック内に於いて、後に続く動作を実行するかどうかの条件を指定する。 式 は、結果としてビット幅１の値を返すものであれば何でも指定でき、結果の値が１なら条件成立・０なら条件不成立となる。

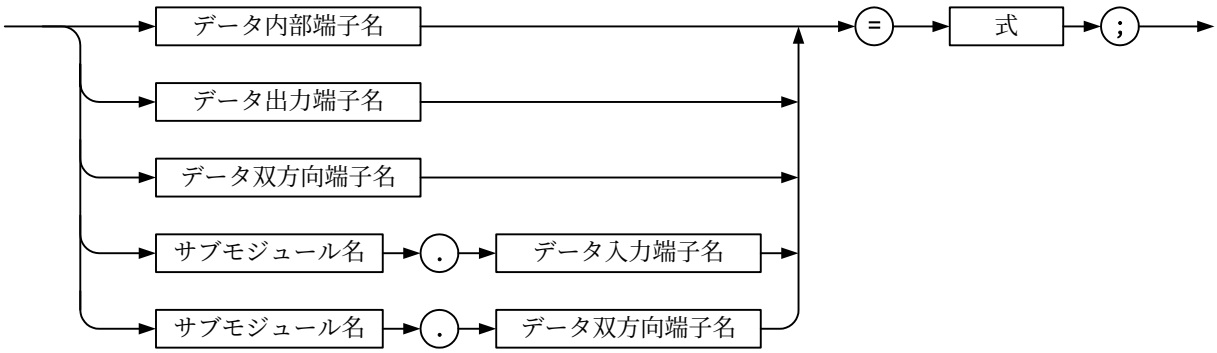
条件として式が評価される時点では、その式が持つ値は不定であってはならない。ただし、条件として評価されないことが確定しているクロック期間内では不定であっても良い。

単位動作



回路動作を定義する。

端子への値の出力

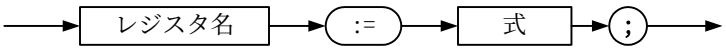


指定されたデータ端子に 式 で指定する値を出力する。指定したデータ端子と 式 の値とはビット幅が一致していなければならない。

データ端子に対する値の出力は、本動作が指示された時点で直ちに実行される。

同一クロック期間内に、異なる値を出力することは許されない。

レジスタへの書き込み

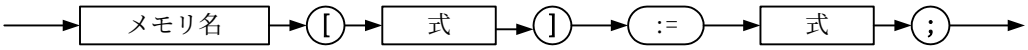


レジスタ名 で指定するレジスタに 式 で指定する値を設定する。指定したレジスタと 式 の値とはビット幅が一致していなければならない。

レジスタに対する値の設定は、本動作が指示された後のクロックの立ち上がり時点で実行される。

同一クロック期間内に、異なる値を設定することは許されない。

メモリへの書き込み



メモリ名 で指定するメモリに値を書き込む。

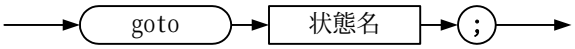
書き込み位置（アドレス）は、［ ］内の 式 で指定し、書き込みデータは 式 （右側）で指定する。指定するアドレスおよびデータのビット幅は、それぞれメモリのビット幅と一致していなければならない。

メモリに対する値の設定は、本動作が指示された後のクロックの立ち上がり時点で実行される。

同一クロック期間内に、異なる値を設定することは許されない。

なお、メモリに関する記述は、機能回路内のみで許される。

状態遷移



ステージまたはセグメントの現在の状態（ステート）を遷移する。

状態の遷移は、本動作が指示された後のクロックの立ち上がり時点で実行される。

同一クロック期間内に、異なる状態への遷移を指示することは許されない。

なお、この動作は、ステート内の 動作 記述としてのみ許される。

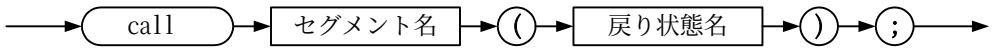
セグメント呼び出し

この動作は、ステージ内ステートまたはセグメントの 動作 記述としてのみ許される。

状態の遷移は、本動作が指示された後のクロックの立ち上がり時点で実行され、指定されたセグメント内の初期状態（ステート）に遷移する。

同一クロック期間内に、異なる状態への遷移を指示することは許されない。

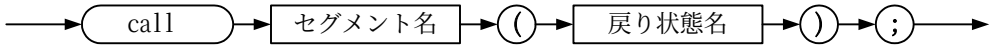
（1）ステージ内ステートからのセグメント呼び出し



ステージの現在の状態を遷移する。

戻り状態名 は、呼び出されたセグメントが セグメントからの復帰 動作を実行したときに遷移先となる（自ステージ内の）ステートを指定する。

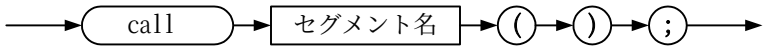
（2）セグメントからの（戻り状態を指定した）セグメント呼び出し



セグメントの現在の状態を遷移する。

戻り状態名 は、呼び出されたセグメントが セグメントからの復帰 動作を実行したときに遷移先となる（呼び出したセグメント内の）ステートを指定する。

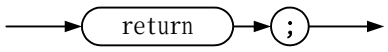
（3）セグメントからの（戻り状態を指定しない）セグメント呼び出し



セグメントの現在の状態を遷移する。

呼び出したセグメントをM・呼び出されたセグメントをＳとするとき、Ｓが セグメントからの復帰 動作を実行したときは、Mが呼び出された際に指定されたステートへ一気に戻る。

セグメントからの復帰



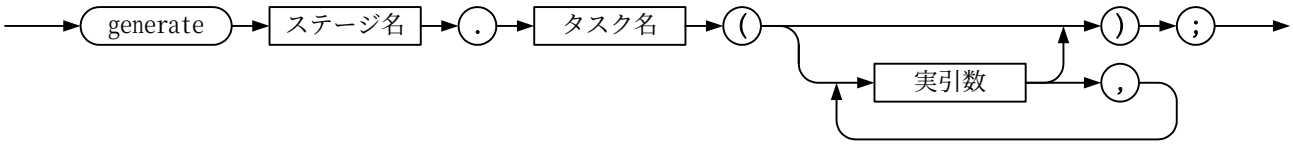
セグメントの現在の状態を遷移する。

状態の遷移は、本動作が指示された後のクロックの立ち上がり時点で実行される。

同一クロック期間内に、異なる状態への遷移を指示することは許されない。

なお、この動作は、セグメントの 動作 記述としてのみ許される。

ジョブの生成

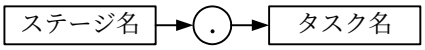


ステージ名 で指定するステージを タスク名 で指定する動作モードで開始させる。ステージ動作の開始は、本動作が指示された後のクロックの立ち上がり時点からである。リセット直後は、ステージの初期状態として指定されたステートが最初の状態となる。ステージのとる状態は、明示的に遷移を指示しない限り変更されない。ステージの停止中も状態は保持されている。

実引数 で指定した値は、タスクを定義した際に仮引数として指定したレジスタに順番に設定される。両者の引数の個数に過不足があってはならない。また、設定される値とレジスタのビット幅は一致していなければならない。

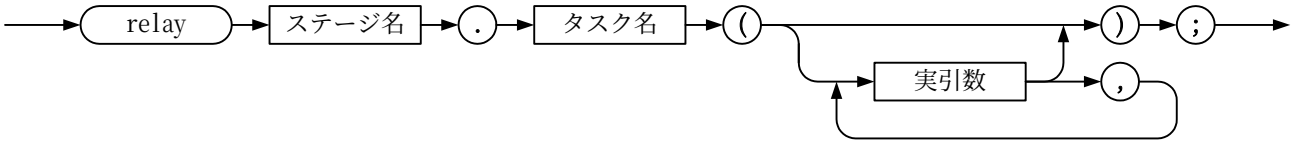
同一のステージに対し、異なるタスクで同時動作させるような指示を行ってはならない。

なお、ステージがこのタスクで動作している期間に於いて、



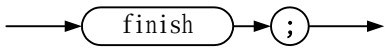
の値は１（ビット幅１）となり、それ以外の場合は０（ビット幅１）となる。

ジョブの転送



動作の内容は、ジョブの生成と同様である。ただし、本動作を指示出来るのはステージ内の 動作 の中だけである。また、本動作を指示したステージは、暗黙に ジョブの終了 動作も同時に行ったと見なされる。

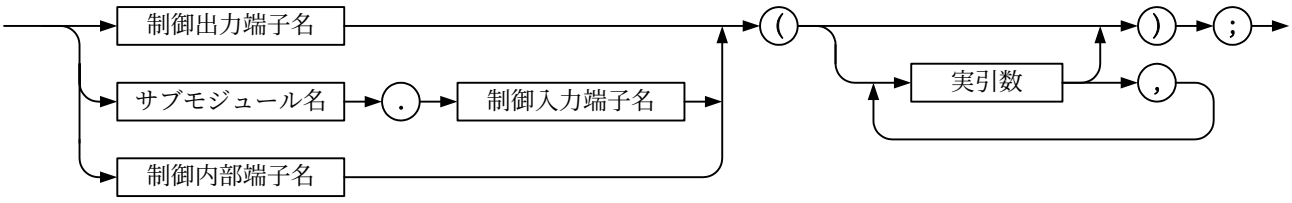
ジョブの終了



本指示を行ったステージの動作を終了する。ステージ動作の終了は、本動作が指示された後のクロックの立ち上がり時点である。

もし、あるステージのジョブの終了とそのステージへのジョブの生成（または転送）が同時に指示された場合は、ジョブの生成（または転送）が優先する。本動作を指示出来るのは、ステージ内の 動作 の中だけである。

制御端子の起動



指定した制御端子を起動する。

（注）「制御端子を起動する」とは、制御端子の値を１にすること。

制御端子は、（データ端子とは違い）値が不定になることはない。起動されていない制御端子の値は必ず０である。また、制御端子のビット幅は１である。

実引数 で指定した値は、「制御端子の仮引数」として指定したデータ端子に順番に設定される。両者の引数の個数に過不足があってはならない。また、設定される値とデータ端子のビット幅は一致していなければならない。

制御端子に対する値１の設定、およびデータ端子に引数値の設定は、本動作が指示された時点で直ちに実行される。同一クロック期間内に、何度起動しても良いが、異なる値を引数とすることは許されない。

この起動動作は、１クロック期間のみ有効であり次のクロック期間に影響は及ばない。（次のクロック期間に起動されない制御端子の値は０である。）

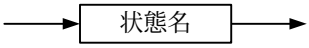
instruct によって、この制御端子の起動時に実行する 動作 が指定されていれば、実行する。

空文

なんの動作もしない。

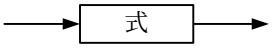
必ず動作記述を置かなければならない場所に、何も行わせたくない場合に使用する。

戻り状態名



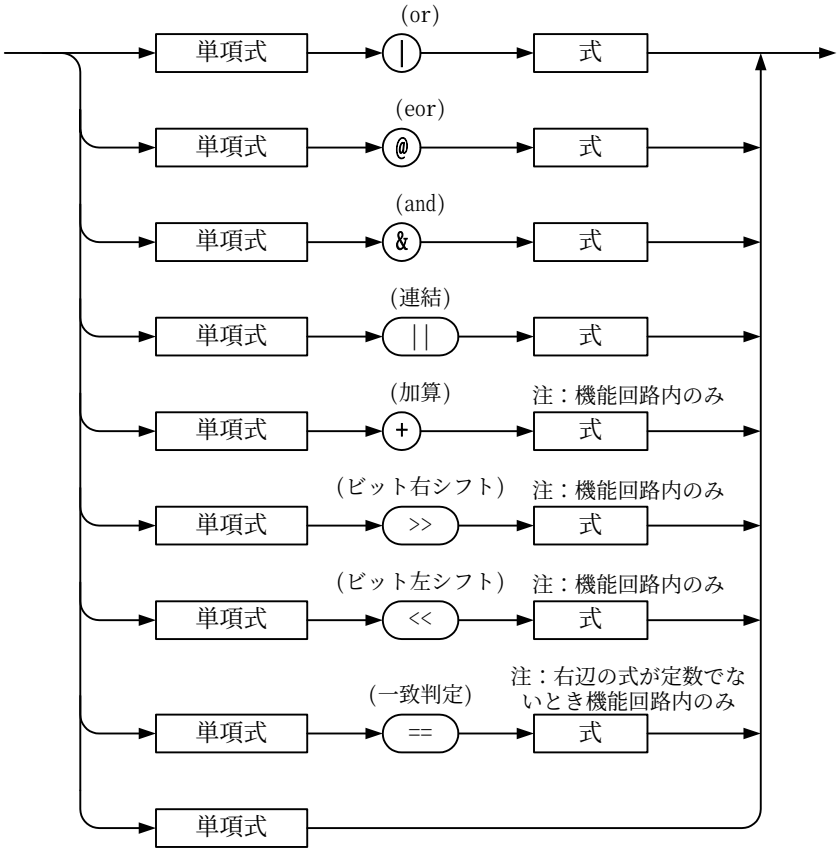
セグメント呼び出しの際に指定する。セグメント呼び出しからの復帰ステートを指定する。

実引数



仮引数として指定されている端子やレジスタに引き渡す値。値のビット幅は、引き渡し先のビット幅と一致していなければならない。

式



ｏｒ演算子　　左右の値（２進数）の論理和をとる。左右の値のビット幅は同じでなければならず、結果のビット幅も同じである。

ｅｏｒ演算子　　左右の値（２進数）の排他的論理和をとる。左右の値のビット幅は同じでなければならず、結果のビット幅も同じである。

ａｎｄ演算子　　左右の値（２進数）の論理積をとる。左右の値のビット幅は同じでなければならず、結果のビット幅も同じである。

連結演算子　　左辺の値（２進数）のビット列のＬＳＢ側に右辺の値（２進数）のビット列を連結する。結果のビット幅は左右両辺の値のビット幅の和になる。

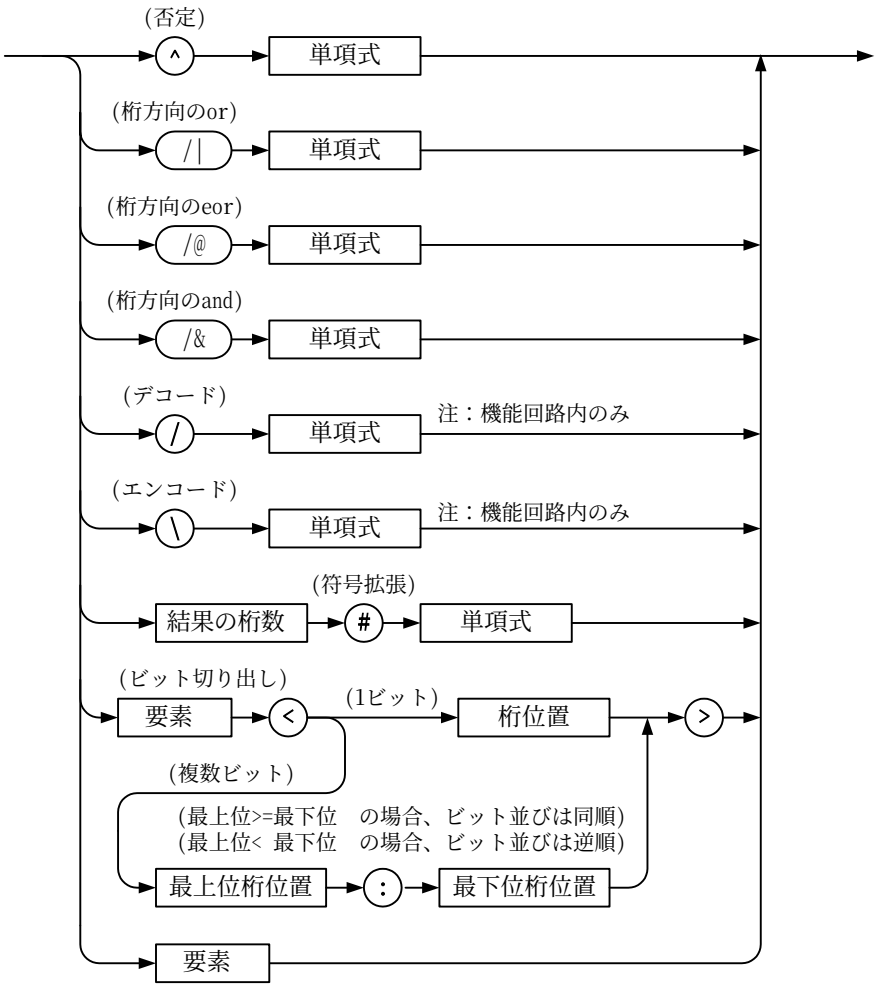
加算演算子　　左右両辺の値の和をとる。結果のビット幅は、左右両辺のビット幅の大きい方と同じになる。桁あふれが起きた場合、最上位ビットは破棄される。

右シフト演算子　　左辺の値（２進数）のビット列を（ＭＳＢ側に０を補充する形で）右辺の値だけ右シフトする。結果のビット幅は左辺と同じになる。

左シフト演算子　　左辺の値（２進数）のビット列を（ＬＳＢ側に０を補充する形で）右辺の値だけ左シフトする。結果のビット幅は左辺と同じになる。

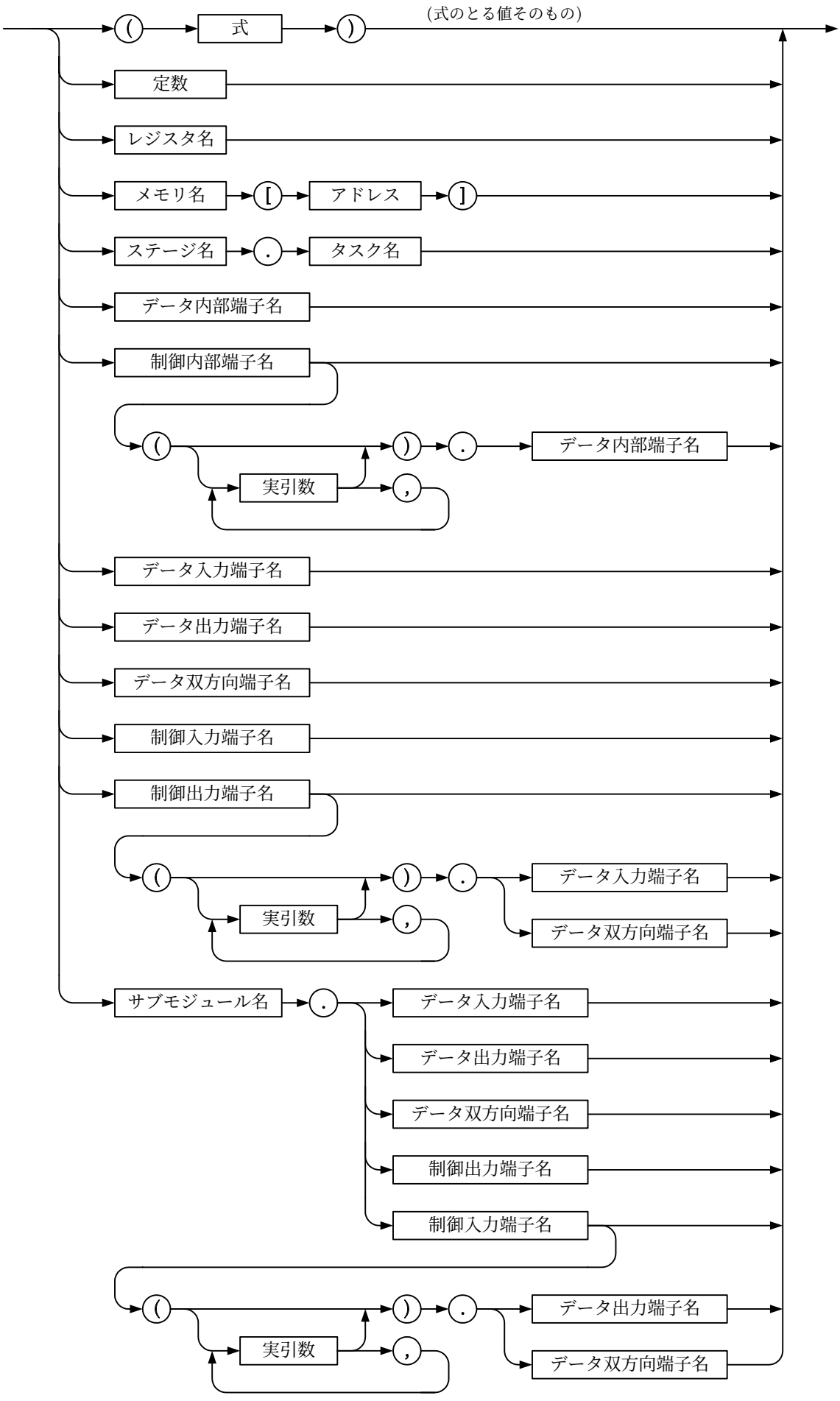
一致判定演算子　　左右の値が同じか判定する。左右の値が同じ場合は１、異なる場合は０となる。左右の値のビット幅は同じでなければならない。結果のビット幅は１である。

単項式



否定 演算子	後に続く値（２進数）をビット反転する。結果のビット幅は元の値と同じである。
桁方向の ｏｒ演算子	後に続く値（２進数）を単なるビット並びと見なして、その論理和をとる。結果のビット幅は１である。
桁方向の ｅｏｒ演算子	後に続く値（２進数）を単なるビット並びと見なして、その排他的論理和をとる。結果のビット幅は１である。
桁方向の ａｎｄ演算子	後に続く値（２進数）を単なるビット並びと見なして、その論理積をとる。結果のビット幅は１である。
デコード 演算子	後に続く値をビット位置と見なして、そのビット位置だけが１となる２進数を生成する。（このとき、ＬＳＢのビット位置を０とする。） 結果のビット幅は、元の値のビット幅をｎとして、２のｎ乗である。
エンコード 演算子	後に続く値（２進数）をＭＳＢ側から探索して、最初に１となるビットの位置番号を結果とする。（このとき、ＬＳＢのビット位置を０とする。） 結果のビット幅は、元の値がビット幅１の場合はビット幅２、それ以外は、ｎを以下の条件を満たす整数として、ビット幅（ｎ＋１）である。 ２の（ｎ－１）乗　＜　元の値のビット幅　≦　２のｎ乗 元の値が０であって、１となるビットがない場合は、結果のＭＳＢが１となる。
符号拡張 演算子	後に続く値（２進数）を指定されたビット幅に拡張する。ビット幅の拡張は、元の値のＭＳＢを必要なビット数分複写して上位桁に追加することで行う。 指定された結果のビット幅が、元の値のビット幅より小さい場合は、上位ビット側を切り捨てる。
ビット切り出し演算子	後に続く値（２進数）から、指定された範囲にあるビットを取り出し、新たな２進数を生成する。 指定されたビット位置が元の値のＭＳＢより上位の場合は、ＭＳＢの上位に必要なビット数だけ（値０のビット）が存在するとみなす。 最上位桁位置の指定値が最下位桁位置の指定値より小さい場合は、両者の指定を入れ替えて得られる結果の２進数のビット並びを逆順にしたものを結果とする。

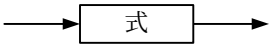
要素


(式のとる値そのもの)
(
式
)
定数
レジスタ名
メモリ名
[
アドレス
]
ステージ名
.
タスク名
データ内部端子名
制御内部端子名
(
)
.
データ内部端子名
実引数
,
データ入力端子名
データ出力端子名
データ双方向端子名
制御入力端子名
制御出力端子名
(
)
.
データ入力端子名
実引数
,
データ双方向端子名
サブモジュール名
.
データ入力端子名
データ出力端子名
データ双方向端子名
制御出力端子名
制御入力端子名
(
)
.
データ出力端子名
実引数
,
データ双方向端子名

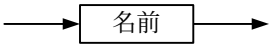
（　）で囲まれた 式 は、式の取る値そのものとなる。

レジスタ，メモリ，端子が指定されている場合は、現時点でそれぞれに設定されている値が結果となる。

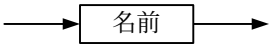
アドレス



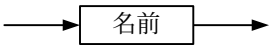
モジュール名



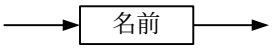
機能回路名



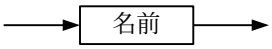
データ入力端子名



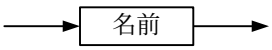
データ出力端子名



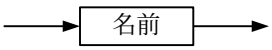
データ双方向端子名



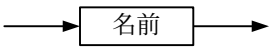
データ内部端子名



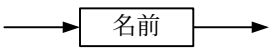
制御入力端子名



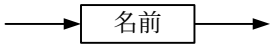
制御出力端子名



制御内部端子名



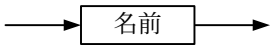
サブモジュール名



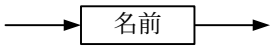
レジスタ名

メモリ名

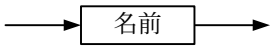
ステージ名



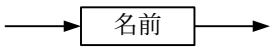
タスク名



セグメント名



状態名



ビット幅

ワード数

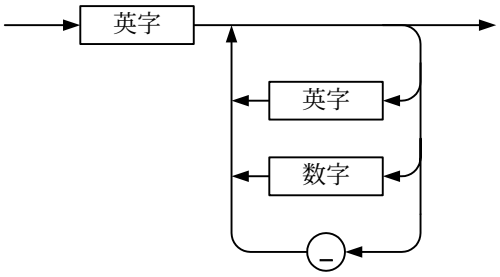
結果の桁数

桁位置

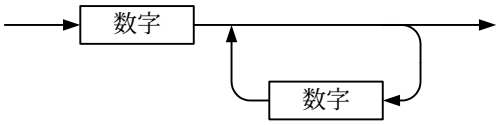
最上位桁位置

最下位桁位置

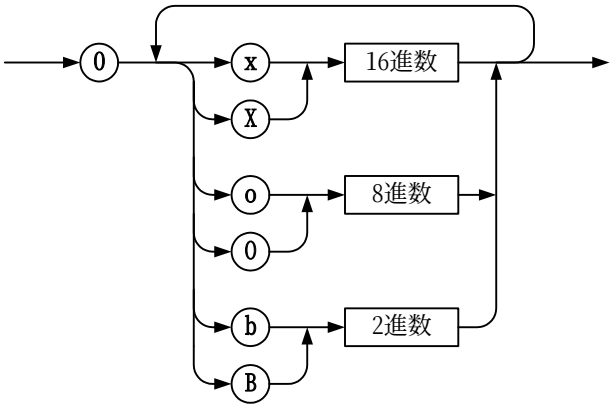
名前



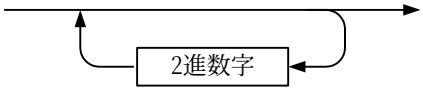
数



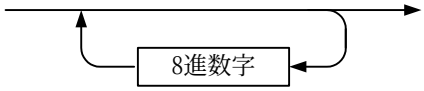
定数



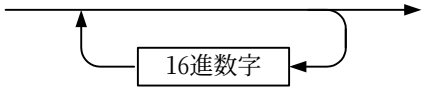
2進数



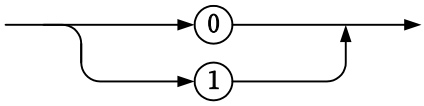
8進数



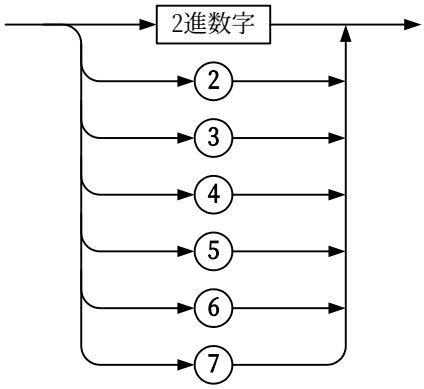
16進数



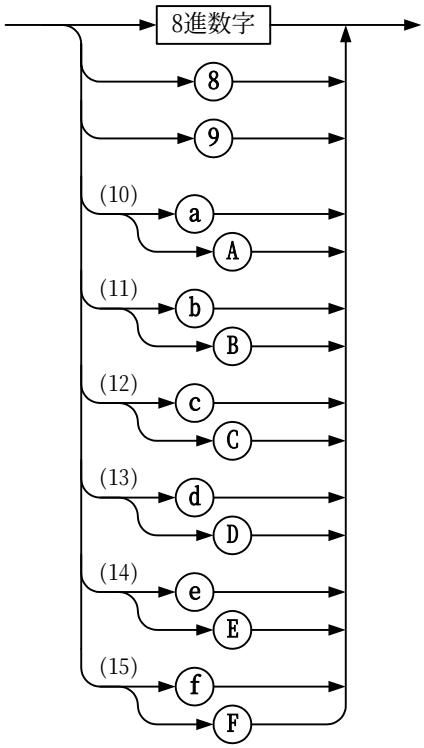
2進数字



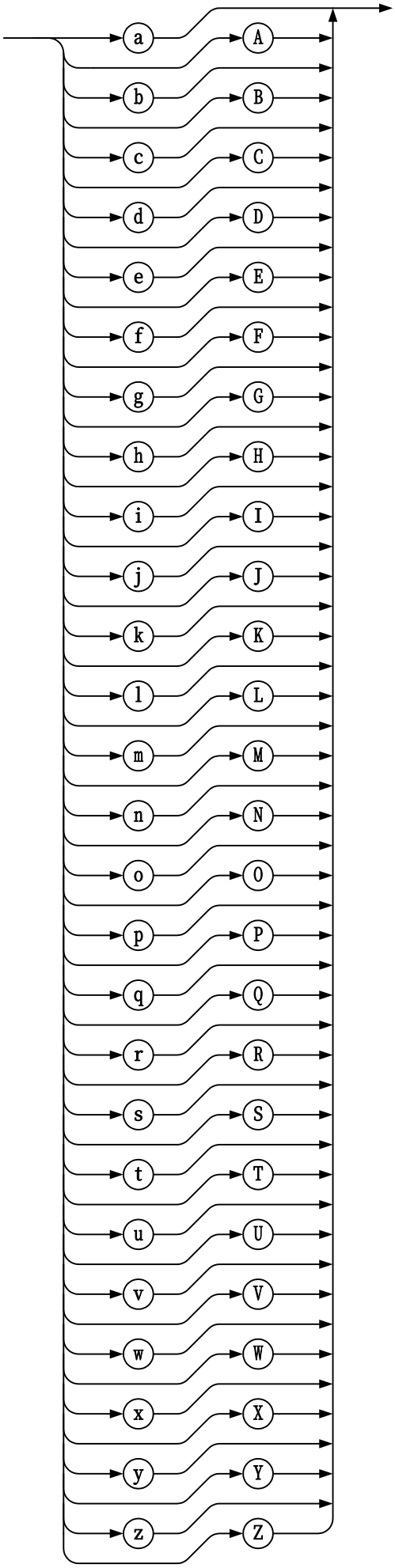
8進数字



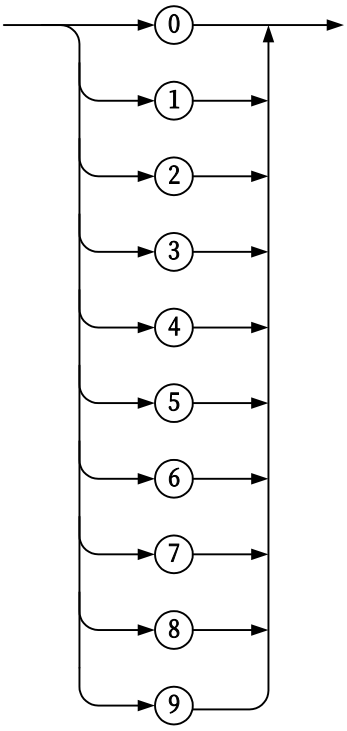
16進数字



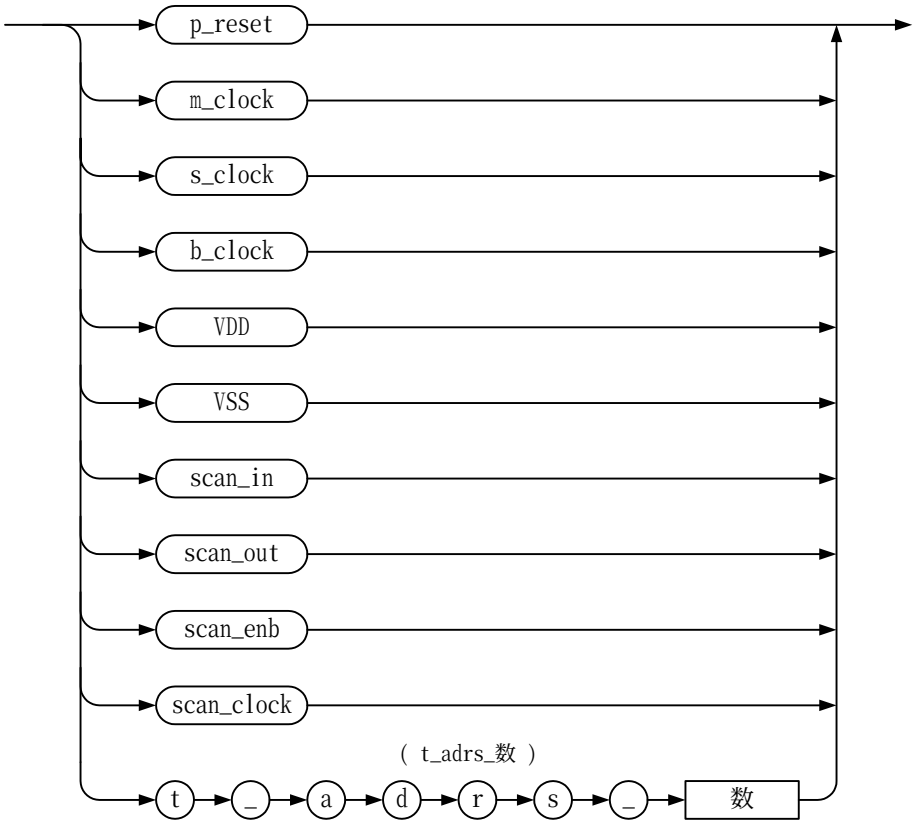
英字


a
A
b
B
c
C
d
D
e
E
f
F
g
G
h
H
i
I
j
J
k
K
l
L
m
M
n
N
o
O
p
P
q
Q
r
R
s
S
t
T
u
U
v
V
w
W
x
X
y
Y
z
Z

数字


0
1
2
3
4
5
6
7
8
9

予約語



ＳＦＬ記述としては使用しないが、ＳＦＬ処理系のためのキーワードとして予約されている。

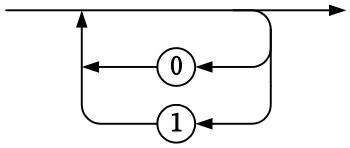
ＳＦＬ記述の中で 名前 として使用してはならない。

おわり

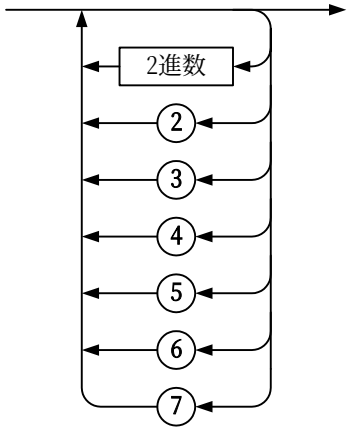
別紙１

・ＳＦＬ仕様書第１版改２までの「2進数」「8進数」「16進数」の構文図。

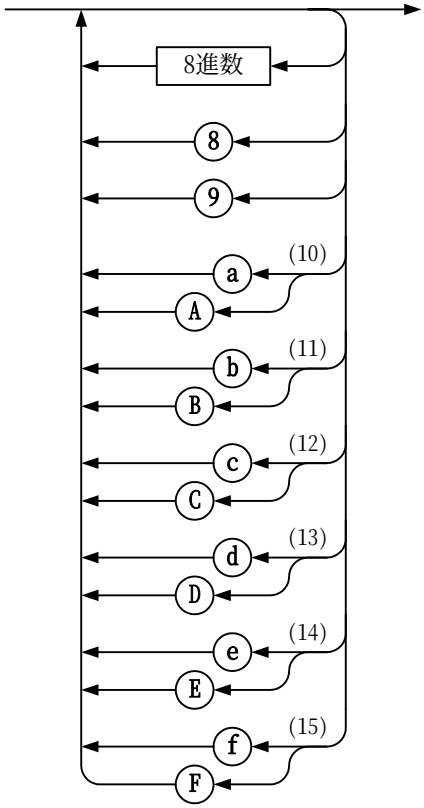
2進数



8進数



16進数



2013年12月21日開催の第39回パルテノン研究会で、この構文図では8進数（および16進数）に於いて「文字無しの連鎖（矢印のみのループ）」が可能となるため、パーサが無限ループに陥るとの指摘がなされ、その修正案が提案・承認された。修正案に基づく新しい構文図は２６ページ参照。