The WAM instruction set

Constraint	Specialized constraint (with execution order and type)	
	Loading argument registers (just before a call)	
$X = Y$	**put_variable** V_n, R_i	Create a new variable, put in V_n and R_i.
$X = Y$	**put_value** V_n, R_i	Move V_n to R_i.
$X = C$	**put_constant** C, R_i	Move the constant C to R_i.
$X = nil$	**put_nil** R_i	Move the constant *nil* to R_i.
$X = F/N$	**put_structure** F/N, R_i	Create the functor F/N, put in R_i.
$X = ./2$	**put_list** R_i	Create a list pointer, put in R_i.
	Unifying with registers (head unification)	
$X = Y$	**get_variable** V_n, R_i	Move R_i to V_n.
$X = Y$	**get_value** V_n, R_i	Unify V_n with R_i.
$X = C$	**get_constant** C, R_i	Unify the constant C with R_i.
$X = nil$	**get_nil** R_i	Unify the constant *nil* with R_i.
$X = F/N$	**get_structure** F/N, R_i	Unify the functor F/N with R_i.
$X = ./2$	**get_list** R_i	Unify a list pointer with R_i.
	Unifying with structure arguments (head unification)	
$X.i = Y$	**unify_variable** V_n	Move next structure argument to V_n.
$X.i = Y$	**unify_value** V_n	Unify V_n with next structure argument.
$X.i = C$	**unify_constant** C	Unify the constant C with next structure argument.
$X.i = nil$	**unify_nil**	Unify the constant *nil* with next structure argument.
$\bigwedge_{1 \le j \le N} X.(i+j-1) = Y_i$	**unify_void** N	Skip next N structure arguments.
	Managing unsafe variables	
$X = Y$	**put_unsafe_value** V_n, R_i	Move V_n to R_i and globalize.
$X.i = Y$	**unify_local_value** V_n	Unify V_n with next structure argument and globalize.
	Procedural control	
	call P, N	Call predicate P, trim environment size to N.
	execute P	Jump to predicate P.
	proceed	Return.
	allocate	Create an environment.
	deallocate	Remove an environment.
	Selecting a clause (conditional branching)	
	switch_on_term V, C, L, S	Four-way jump on type of A_1.
	switch_on_constant N, T	Hashed jump (size N table at T) on constant in A_1.
	switch_on_structure N, T	Hashed jump (size N table at T) on structure in A_1.
	Backtracking (choice point management)	
	try_me_else L	Create choice point to L, then fall through.
	retry_me_else L	Change retry address to L, then fall through.
	trust_me_else *fail*	Remove top-most choice point, then fall through.
	try L	Create choice point, then jump to L.
	retry L	Change retry address, then jump to L.
	trust L	Remove top-most choice point, then jump to L.