This has a description with the UCI options for IvanHoe currently (999946-Beta).

**Option: Hash**
This takes values from 1 to 65536 in megabytes, though requires too values to regard height bits. We cannot put this information on the option itself for the various GUI insists we determine it as "Hash". Warning: upon a value who is too large, the access will be slowed.

**Option: PawnsHash**
This takes values from 1 to 1024 in megabytes, and only works for binary powers (e.g: 1, 2, 4, 8, 16, 32, 64, 128,…). The name is wrong, as king location is also in pawns. Rising this can lead to more hash hits with pawn evaluation. Again making it too large will slow the access in the memory system.

**Option**: **PawnsHashOneEighth**
This automatically sets PawnsHash to 1/8th of Hash.

**Option: PVHash**
This takes values from 1 to 1024 in megabytes, and only works for binary powers. The name is wrong, as king location is also in pawns. Increase this if TryPVinAnalysis is off and the PVs are too short.

**Option: EvalHash**
This takes values from 1 to 1024 in kilobytes!, and only works for binary powers. Do not apply this too large! Enough to reach large pages has sufficience.

Personal Advise: 1024 Hash, 128 PawnsHash, 128 PVHash, 2048kb EvalHash (large).

**Option: Threads**
This counts the multicore CPUs. The current maximum is 8 as the SMP workings are untested for more. CPUs are automatically detected.

**Option: Ponder**
This demands to the UCI whether to ponder. We have tested Ponder and it now works as intended. So fast moves are played upon desire from ponderhit. They can be bad occasionally, but all is true in generally. The Easy Move logic sees our attention here, where we think are problems.

**Option: VerifyNullMove**
This verifies null move. The default is true as that is better. Turning this off cannot see much gain and loses in zugzwang.

**Option: AlternativeTimeUsage**
This turns on the AlternativeTimeUsage. We have made no thorough tests for the application here.

**Option: AllowInstantMoveFromHash**
This allows instant moves to be made from hash when in ponder off mode. The value of this is when a move is clear.

**Option: BufferTime** (milliseconds)
This lists in milliseconds how much time to borrow. This will not be used. For a 1 minute flat game, 1000 millisecond (1 second) is OK.

**Option: OutputDelay** (milliseconds)
This lists in milliseconds how long for waiting until output is emitted.

**Option: MultiCentiPawnPV**
This limits the gap of worse moves with MultiPV. For making this 100 centipawns will eliminate moves worse than that much behind.

**Option: RandomCount**
This turns on the randomizer effect unless it is zero. The number of random numbers to add is controlled here. The maximum is 8.

**Option: RandomBits**
This determines how much bits to use for each random component. The value can be 1 or 2 or 3. With one random bit, the value is -1 or 0 or 1, and with 2 it has from -3 up to 3, and with 3 it has from -7 to 7. Each random is added according to the count of RandomCount. To see this the value of RandomCount is 4 and RandomBits is 2 should add 4 values from -3 to 3 for each evaluation. We use RandomCount as 4 or 8 and RandomBits as 1 for testing.

**Option: UCI_White_Bishop_Pair_Scale** (cp)
**Option: UCI_White_Pawn_Scale** (cp)
**Option: UCI_White_Knight_Scale** (cp)
**Option: UCI_White_Light_Bishop**_Scale (cp)
**Option: UCI_White_Dark_Bishop**_Scale (cp)
**Option: UCI_White_Rook_Scale** (cp)
**Option: UCI_White_Queen_Scale** (cp)
**Option: UCI_Black_Bishop_Pair_Scale** (cp)
**Option: UCI_Black_Pawn_Scale** (cp)
**Option: UCI_Black_Knight_Scale** (cp)
**Option: UCI_Black_Light_Bishop_Scale** (cp)
**Option: UCI_Black_Dark_Bishop_Scale** (cp)
**Option: UCI_Black_Rook_Scale** (cp)
**Option: UCI_Black_Queen_Scale** (cp)

These are user fun options with rescaling pieces. We have not applied them internally.

**Option: MaterialWeighting**
**Option: KingSafetyWeighting**
**Option: PawnsWeighting**
**Option: StaticWeighting**
**Option: MobilityWeighting**

These are more user fun options for rescaling. The units are all in 1024s.

**Option: AlwaysAnalyze**
This option allows GUI compatibility for some and propels the companion of MultiPV mode when playing a game.

**Option: TryPVInAnalysis**
This option demands an expansion of the PV in analysis mode. If this is not on the PV can be truncated from hash hits.

**Option: FixedAgeAnalysis**
This option delimits the AGE count in the hash table when in analysis. This is useful when applying forward and backward analysis as in the contrary, the AGE is incremented upon every position from the GUI. However you need to be careful and apply "ucinewgame" to shoal the hash when the higher depths entries become boggy, or when applying an independent position.

**Option: SendCurrmove**
This option demands the currmove to have been sent even when game mode is on.

**Option: DoHashFull**
This option implies to send hashfull at the second updates. There is a little overhead.

**Option: GetFEN**
This utility endeavors for IvanHoe to compute the FEN as an info string in UCI.

**Option: TimeImitateOpponent**
**Option: TimeMoreWhenLosing**
**Option: TimeMoreWhenWinning**
**Option: TimeEasyFactor**
**Option: TimeEasyFactorPonder**
**Option: TimeBattleFactor**
**Option: TimeOrdinaryFactor**
**Option: TimeAbsolutePercent**
**Option: TimeDesiredMillis**
**Option: TimeBookExitMoves**

These options control the standard time usage. Our defaults seem decent. The DesiredMillis sets up how much time to use, except in movestogo mode when that is apparent. The default 40 uses 40/1000 or 1/25 of the time back in the desired time. The factors then say how much of the desired time to use in situations. The absolute percent caps the amount that can be used in worst scenarios. The book exit moves demands to use extra time on the first moves when the book was left. We put this as 0 and the matter is not much.

**Option: ExtraExtendInCheck**
This option when on will extend an extra half-ply when in check. The default is off.

**Option: SplitAtCUT, SplitDepthCUT, SplitDepthALL, SplitDepthPV**
These options appear with -DUSER_SPLIT, and allow the user to control the multicore mode for more.

RobboBase Options**:**

**Option: RobboBaseDynamicLibraryFile**
This  is for libraries .so (Linux) and .dll (Windows). It should only be modified when the library file is located elsewhere other than the engine's current working directory.

**Option: RobboTripleBaseDirectory**
This sets the RobboTripleBase directory to a string, and loads the TripleBases.

**Option: UnloadRobboTripleBases**
This unloads the RobboTripleBases from the directory that is set.

**Option: RobboTotalBaseDirectory**
This sets the RobboTotalBase directory to a string, and loads the TotalBases.

**Option: DeregisterRobboTotalBases**
This deregisters the RobboTotalBases from the directory that is set.

**Option: RobboTotalBaseCacheSize**
This sets the RobboTotalBase cache size. The values are 1 to 1024 megabytes in binary. The value should be minimal for the TotalBase use finds itself only at root positions. The exception is when building the TotalBase lot, though that access is separate. So 1 megabyte seems fine for play and analyzing here.

**Option: DynamicLoadTripleBaseCacheSize**
This sets the size for dynamic Triple bases in cache. Unless said all with 5 pieces or more go here. The size depends on your work. With analysis of endgames, 256MB or more counts wise.

**Option: TripleHashSize**
This takes values from 1 to 4096 in megabytes, and only works for binary powers. It stores score for TripleBase positions probed which complements DynamicLoadTripleBaseCacheSize**.** It can however decrease NPS when set too high.

**Option: TripleWeakProbeDepthHalfPly**
This sets the depth limit to weak probes. When depth exceeds it, a weak probe is incurred. Negative values are possible too, and can have value. Increasing too large turns off TripleBases in essence. Weak probes are not handily free in cost, though worry is low generally. There is also a plus with statistics in having weak probes attend.

**Option: TripleDefiniteProbeDepthHalfPly**
This sets the depth limit with TripleBase probes. When depth exceeds it, a TripleBase probe is incurred. Probe from disk will attend much time. Guideline: 40 half ply sums 20 ply, at 1 millionth of tree if split binary. Note: Probe still can technically fail if cache tendency is arraigned (SMP defect).

**Option: TripleDefiniteProbeHeight**
This sets the height for definite TripleBase probes. When height is as small as value, definite TripleBase probe is met. The control is switched for more at root. This complements your above option.

**Option: LoadOnWeakProbe**
This sets whether to load TripleBases in the background during a search when a probe is made. If this is not set, your TripleBases must be in memory to function.

**Option: RobboTripleBulkLoadThisDirectory**
**Option: RobboTripleBulkDetachThisDirectory**
This moves an entire directory of RobboTripleBases into memory (RAM). The cache is not filled by these. The 5s fill over 500MB. The directory should be minor, not root. Example is "value 5" or "value 345Z" for those. Multiple directories are separated using pipe (|). It can also be "value /media/disk/RobboTripleBase/5" or "value /media/disk/RobboTripleBase/345Z" using full path.

**Option: RobboTripleBulkLoadThisName**
**Option: RobboTripleBulkDetachThisName**
This moves one file for RobboTripleBases to be used from RAM other than disk. There is care, as TripleBases use recurrings to call them more. The situation send a time in difficult. Unless there is necessary, the WeakProbe access deems enough.

**Option: MultiPV**
This sets the MultiPV number. This will work in game mode additionally if AlwaysAnalze is made too.

**Option: UCI_Chess960**
This allows support for Fischer Random Chess. Some GUIs handle it like ponder.

Optional options:
**DebugTimeManagement**: to appear for debugging with time management

Further utilities in -DUTILITIES:
**eval**: lists an evaluation of the position, but still not prepared finally
**benchmark** [go string]: achieve a benchmark upon 16 positions, with UCI string appended seeing the default for "go movetime 1000"
**perft** [n]: count a perft for the position to n moves
**perft-check** [n] [c]: count a perft for the position to n moves and ensure the answer is c
**drawboard**: draws a board in the style of Crafty
**verify-tripl**e [a] [b] [c] [d]: utility internal function for the verifier with triple bases which need the loading and the TotalBase registry, and notation here is the internal accounting of pieces so [7] [14] [9] [0] is wQ against bR+bP

The ZOG MP does not turn on unless you compile it and then it hangs.

MonteCarlo:

UCI Version: "**go montecarlo** [options]"
Options:
  cpus # : default has 1
  min # : default has -15000
  max # : default has 15000
  length # : default has 65535
  depth # : default has 9
  moves [list]
This searches the moves with the moves [list] who comes last, churning it on [cpus #] of cpus. Each iteration runs for [length #] ply at each move searching [depth #] ply. When score exceeds [max #] or recedes [min #] the termination occurs too. The default is 9 ply searches and no termination until game end.

**position** fen 4r3/4b3/3p1k2/2pP4/2P5/1P5K/6R1/6N1 w - - 0 0
**go montecarlo** cpus 4 min -25 max 325 length 40 depth 10 moves g2a2 g2e2 g2f2 g1f3 g2g3

The output is "MCresult [move] [score] [cpu id]" as
MCresult g2g3 64 0
MCresult g2e2 99 3
MCresult g2f2 -30 1

The ComradesGUI shall decide it easier to access. The internal move selector from [list] is now random. To be better, weight upon results.