Algorithm Specification – VIC State Updating

Version 3.0.0

June 23, 2015

Table of Contents

Record of Changes

Version	Date	Description of Change
1.0.0	May 14, 2015	Version 1 draft specification
2.0.0	June 3, 2015	Major revision of original specification
2.0.1	June 5, 2015	Updated logic for categorical constraints for SPEC-5
3.0.0	June 18, 2015	This update attempts to reduce the complexity of the update process, but at the expense of decreased simulation realism, Particular changes include: updating is done by HRU (not by band); expanded from 3 cases to six cases; simplified mass-balance updating (SPEC5) considerably; all tables contain default state values

List of Symbols

Symbols	Description
	Variables
A	HRU or band area (as fraction of grid cell)
B	Number of elevation bands per cell
H	Number of HRUs per cell
Ω	State variable (generic)
	Index Variables
b	Band index
c	Cell ID
h	HRU index
t	Initial model state index (i.e. prior to updating)
$t*$	Final model state index (i.e. after updating)
	Subscripts
g	Glacier HRU
op	Open ground HRU
v	Vegetated HRU (i.e. not glacier or open ground)

1 Background

This specification details the method for updating the VIC state file following glacier updating. One of the main features in the coupling of the VIC model to the UBC Regional Glaciation model (RGM) is the feedback of glacier area and surface elevation from the RGM to VIC. Changes in glacier area (passed from RGM to VIC as an updated glacier mask) and surface elevation are incorporated into the VIC model via updating of the vegetation parameter file and the elevation band file. A side-effect of this updating step is the need to adjust certain state variables to ensure conservation of mass and energy within the individual VIC cells as a result of area and elevation updating. Conceptually, this entails the redistribution of water and energy within individual HRUs. For example, the goal of water re-distribution between hydrologic response units (HRUs) is to conserve the volume of water within a grid cell. In general, the following must hold for a given cell:

$$\sum_{h=1}^{H(t)} \Omega(h,t) \cdot A(t) = \sum_{h=1}^{H(t^*)} \Omega(h,t^*) \cdot A(h,t^*) \tag{1}$$

where H is the number of HRUs in the given VIC cell, Ω is a state variable (e.g. water equivalent depth of snow), A is the area fraction of HRU number h, and t and t^* represent the model state before and after glacier updating, respectively.

The specifications described in the following sections for state updating is considered within the context of the pseudo-code shown in the text Box 1 below. The pseudo-code describes an algorithm for looping through cells and HRUs, and then checking for various situations (or cases) that describe the context under which an HRU can be updated. Five of these cases have been indicated (which should cover all possibilities).

The state variables being considered by this specification are summarized in Table 1 through Table 3. The state variables listed in Table 1 are considered mandatory for proper model check pointing. Table 2 list state variables that are considered miscellaneous at this time; they are not actually necessary for proper state updating (their inclusion in the state file is simply an artefact of earlier programming efforts) and they may be dropped from future implementations. Table 3 list state variables that are only required when certain code options are selected (i.e. *DIST_PRCP*, *EXCESS_ICE*, *LAKES*, *SPATIAL_FROST* and *SPATIAL_SNOW*). However, as these code options are currently untested and their use discouraged, these state variables are not explicitly updated.

The pseudo-code algorithm provides specifications for updating VIC cell and HRU metadata state variables and identifies five cases requiring unique specifications for updating HRU state variables:

1) Case 1: In this situation a new HRU appears, either in an existing elevation band (GLACEIR or OPEN) or in a new elevation band (GLACIER).
2) Case 2: This is the trivial case wherein the area of the HRU h does not change between states t and t^*, hence no state updating needs to occur.
3) Case 3: This is the case wherein the area of HRU h changes ($\Delta A(h,t^*)\neq 0$), but the HRU persists in both states t and t^*. In this situation state is updated within the HRU

4) Case 4: In this case an HRU disappears but the elevation band persists, either because a GLACIER HRU shrinks (and disappears), or expands (and neighbouring OPEN and/or VEG HRUs disappear). In this situation state must be transferred from the disappearing HRU to one of the remaining HRUs in the same elevation band *b*.
5) Case 5: This situation is similar to Case 4, but in this particular case a GLACIER HRU disappears and the elevation band *b* along with it. In this situation state must be transferred to one of the remaining HRUs in the elevation band below, *b*-1. The target HRU in band *b*-1 is prioritized, such that a GLACIER is chosen first, if no GLACIER HRU exists, then the OPEN-GROUND HRU is chosen, and if no OPEN-GROUND HRU exists, then the largest VEGETATED HRU is chosen. Note that it is impossible for a grid cell to have zero elevation bands as there must always be at least one band per cell for the cell to exist. Hence, a band can only disappear if there is an existing band below it (i.e. the lowest elevation band can never disappear).

Note that multiple cases can occur in the same VIC cell and or elevation band, and that an HRU may be subject to several updating operations as certain cases can occur simultaneously. As much as is practicable, the specifications that follow were written to be independent of the order-of-operations. However, the specific requirement to conserve the mass and energy of disappearing HRUs (by 'adding' to neighbouring HRUs) forces some dependency.

```
#Loop through VIC cells and HRUs
for c in NUM_CELLS:

    #Initialize new HRUs and missing values for existing HRUs (ensure initialization before other cases)
    for h in NUM_HRU[c,t*]:
        if (A[c,h,t*]>0 and (A[c,h,t] == 0 or A[c,h,t]  is undefined)):
            do CASE1 #set default values for all state variables
        do Set 'missing' values to defaults #For Mandatory variables only

    #Update State
    for h in NUM_HRU[c,t*]:
        BAND_DISAPPEARS = FALSE
        GLACIER_IN_LOWER_BAND = FALSE
        OPEN_ IN_LOWER_BAND = FALSE

        #Get band index and calculate area change for current HRU
        b = band_index[c,h]
        deltaA = A[c,h,t*] - A[c,h,t]

        if (Aband[c,b,t*]==0 and Aband[c,b,t]>0)
            BAND_DISAPPEARS = TRUE

        if veg_index[c,b,h]==glacier_veg_class:
            GLACIER=TRUE
        else:
            GLACIER=FALSE

        #The intent of the next statements is to determine if a GLACIER or OPEN HRUs exist in the next lower band
        if glacier_veg_class in vegetation_indexes[c,b-1,t*]: GLACIER_IN_LOWER_BAND = TRUE
        if open_ground_veg_class in vegetation_indexes[c,b-1,t*]: OPEN_IN_LOWER_BAND = TRUE

        if deltaA==0:
            do CASE2 #trivial case; no change in area, therefore no change in state values
        elif A[c,h,t*]>0: #Change in area implied (i.e. deltaA<>0) and HRU exists at current state
            if not NEW_HRU:
                do CASE3  #HRU h exists at current and previous state
            else:
                skip to next HRU #New HRU already initialized; skip to next HRU
        else: #Change in area implied (i.e. deltaA<>0), and also implied that HRU no longer exists at current
            if not BAND_DISAPPEARS: #CASE 4 - HRU h disappears but band b remains
                if GLACIER:
                    do CASE4a  #GLACIER disappears - implies OPEN expanding; add state to OPEN
                else:
                    do CASE4b  #non-GLACIER disappears - implies GLACIER expanding; add state to GLACIER
            else: #CASE 5 - Both HRU h and band b disappear - in this case h can only be a GLACIER
                if GLACIER_IN_LOWER_BAND:
                    do CASE5a #add state to GLACIER HRU in band b-1
                elif OPEN_IN_LOWER_BAND:
                    do CASE5b #add state to OPEN HRU in band b-1
                else:
                    do CASE5c #add state to largest VEGETATED HRU in band b-1

        do SANITY_CHECK
```

Box 1. Pseudo-code for VIC state updating

Table 1. Summary of state variables requiring mandatory updating, including default values

State Variable	Description	Default Value
	Cell Metadata	
lat	Grid cell centre latitude	*LAT*
lon	Grid cell centre longitude	*LON*
GLAC_MASS_BALANCE_INFO	Cell ID & mass balance polynomial terms and error	[0,0,0,0,0]
GRID_CELL	Grid cell ID number	*cellID*
NUM_BANDS	Number of bands (set in global file)	*B*
NUM_GLAC_MASS_BALANCE_INFO_TERMS	Number of glacier mass balance terms	5
SOIL_DZ_NODE	Soil thermal node deltas	*SOIL_DZ_NODE*(*h*-1)
SOIL_ZSUM_NODE	Soil thermal node depths	*SOIL_ZSUM_NODE*(*h*-1)
VEG_TYPE_NUM	Number of HRUs in grid cell	*H*(*c*,*t**)
	HRU Metadata	
HRU_BAND_INDEX	Band index	*b*
HRU_VEG_INDEX	HRU vegetation class	*vegIndex*(*h*)
	HRU Water Balance	
LAYER_ICE_CONTENT	Ice content in each soil layer [*SPATIAL_FROST* = FALSE]	0
LAYER_MOIST	Total soil moisture in each layer	0
HRU_VEG_VAR_DEW	Water stored on surface/vegetation	0
SNOW_CANOPY	Snow stored in the canopy	0
SNOW_DENSITY	Snow density	0
SNOW_DEPTH	Snow depth	0
SNOW_PACK_WATER	Water stored in snow pack layer	0
SNOW_SURF_WATER	Water stored in snow surface layer	0
SNOW_SWQ	Total snow water equivalent	0
	HRU Glacier Water Storage	
GLAC_WATER_STORAGE	Water stored in the glacier	0
	HRU Glacier Mass Balance	
GLAC_CUM_MASS_BALANCE	Glacier cumulative mass balance	0
	HRU Snow Pack, Glacier and Soil Energy	
ENERGY_T	Soil temperature at each soil node	0
ENERGY_TFOLIAGE	Vegetation temperature	0
GLAC_SURF_TEMP	Temperature of glacier surface Layer	0
SNOW_COLD_CONTENT	Cold content of snow surface layer	0
SNOW_PACK_TEMP	Temperature of snow pack layer	0
SNOW_SURF_TEMP	Temperature of snow surface layer	0
	HRU Snow Surface Properties	
SNOW_ALBEDO	Albedo of snow	0
SNOW_LAST_SNOW	Days since last snowfall	0
SNOW_MELTING	Snow melting flag [TRUE or FALSE]	"FALSE"
	HRU Program Terms	
ENERGY_TCANOPY_FBCOUNT	*TCANOPY* fallback count	0
ENERGY_T_FBCOUNT	*T* fallback count	0

State Variable	Description	Default Value
ENERGY_TFOLIAGE_FBCOUNT	*TFOLIAGE* fallback count	0
ENERGY_TSURF_FBCOUNT	*TSURF* fallback count	0
GLAC_SURF_TEMP_FBCOUNT	*GLAC_SURF_TEMP* fallback count	0
SNOW_SURF_TEMP_FBCOUNT	*SNOW_SURF_TEMP* fallback count	0

Table 2. Summary of miscellaneous state variables with default values

State Variable	Description	Default Value
GLAC_QNET	Glacier surface net energy balance	0
GLAC_SURF_TEMP_FBFLAG	*GLAC_SURF_TEMP* fallback flag	0
GLAC_VAPOR_FLUX	Glacier vapor flux	0
NONE	???	0
SNOW_CANOPY_ALBEDO	Albedo of snow stored in the canopy	0
SNOW_SURFACE_FLUX	Sublimation from blowing snow	0
SNOW_SURF_TEMP_FBFLAG	*SNOW_SURF_TEMP* fallback flag	0
SNOW_TMP_INT_STORAGE	Temporary canopy interception storage	0
SNOW_VAPOR_FLUX	Snow evaporation and sublimation	0

Table 3. Summary of deferred state variables (for untested code paths) with default values

State Variable	Description	Default Value
	Option.DIST_PRCP = TRUE	
PRCP_MU	Fraction of grid cell that receives precipitation	1
INIT_STILL_STORM	Storm continuity flag [*TRUE* or *FALSE*]	"FALSE"
INIT_DRY_TIME	Time since last storm	0
	EXCESS_ICE = TRUE	
SOIL_DEPTH	Soil moisture layer depths	0
SOIL_EFFECTIVE_POROSITY	Soil porosity when soil pores expanded due to excess ground ice for each soil layer	0
SOIL_DP	Soil damping depth	0
SOL_MIN_DEPTH	Soil layer depth as given in the soil file	0
SOIL_POROSITY_NODE	Soil porosity at each node	0
SOIL_EFFECTIVE_POROSITY_NODE	Soil porosity when soil pores expanded due to excess ground ice for each soil thermal node	0
SOIL_SUBSIDENCE	Subsidence of soil layer	0
	Option.LAKES = TRUE[a]	
LAKE_LAYER_MOIST	Total soil moisture in each layer	0
LAKE_LAYER_SOIL_ICE	Ice content in each soil layer [*SPATIAL_FROST* = TRUE]	0
LAKE_LAYER_ICE_CONTENT	Ice content in each soil layer [*SPATIAL_FROST* = FALSE]	0
LAKE_SNOW_LAST_SNOW	Days since last snowfall	0
LAKE_SNOW_MELTING	Snow melting flag [TRUE or FALSE]	"FALSE"
LAKE_SNOW_COVERAGE	Snow coverage fraction	0
LAKE_SNOW_SWQ	Total snow water equivalent	0
LAKE_SNOW_SURF_TEMP	Temperature of surface snow layer	0
LAKE_SNOW_SURF_WATER	Water stored in snow surface layer	0
LAKE_SNOW_PACK_TEMP	Temperature of pack snow layer	0
LAKE_SNOW_PACK_WATER	Water stored in snow pack layer	0
LAKE_SNOW_DENSITY	Snow density	0
LAKE_SNOW_COLD_CONTENT	Cold content of snow surface layer	0
LAKE_SNOW_CANOPY	Snow stored in the canopy	0
LAKE_ENERGY_T	Soil temperature at each soil node	0
LAKE_ACTIVENOD	Number of nodes whose corresponding layers contain water	0
LAKE_DZ	Thickness of all water layers below surface layer	0
LAKE_SURFDZ	Thickness of surface (top) water layer	0
LAKE_LDEPTH	Depth of liquid water in lake	0
LAKE_SURFACE	Horizontal x-section area at each lake node	0
LAKE_SAREA	Lake surface area of (ice + liquid)	0
LAKE_VOLUME	Lake water volume (including w.e. of lake ice)	0
LAKE_TEMP	Lake water temperature at each node	0
LAKE_TEMPAVG	Average water temperature of entire lake	0
LAKE_AREAI	Area of ice coverage at beginning of time step	0
LAKE_NEW_ICE_AREA	Area of ice coverage at end of time step	0

[a] Many of the *LAKE* state variables are redundant

LAKE_ICE_WATER_EQ	Water equivalent of lake ice	0
LAKE_HICE	Height of lake ice at thickest point	0
LAKE_TEMPI	Lake ice temperature	0
LAKE_SWE	Water equivalence of lake snow cover	0
LAKE_SURF_TEMP	Temperature of surface snow layer	0
LAKE_PACK_TEMP	Temperature of pack snow layer	0
LAKE_SALBEDO	Albedo of lake snow	0
LAKE_SDEPTH	Depth of snow on top of ice	0
	SPATIAL_FROST = TRUE	
LAYER_SOIL_ICE	Ice content of the frozen soil sublayer	0
	SPATIAL_SNOW = TRUE	
SOIL_DEPTH_FULL_SNOW_COVER	Minimum depth for full snow cover	0
SNOW_COVERAGE	Snow coverage fraction	0

2 General Specifications

This specification will often distinguish between *GLACIER*, *OPEN* (i.e. bare soil) and *VEGETATED* Hydrologic Response Units (HRUs). *GLACIER* and *OPEN* HRUs are explicitly identified by land cover classification; this is done by the user in the global parameter file. By inference, *VEGETATED* HRUs are all land cover classes other than *GLACIER* or *OPEN* classes. Note that the *OPEN* class is explicitly designated by the user using by selecting an entry from the vegetation library file, and it is not to be confused with the VIC model's default bare soil land cover classification.

2.1 Cell and HRU Metadata

Generally cell and HRU metadata values will remain unchanged between state *t* and *t**, except for new HRUs. Hence the spec for generic cell metadata state variable Ω for HRU *h* is

CASE 1	$\Omega(h,t^*) = default\ values$
CASE 2	$\Omega(h,t^*) = \Omega(h,t)$
CASE 3	$\Omega(h,t^*) = \Omega(h,t)$
CASE 4	$\Omega(h,t^*) = 0$
CASE 5	$\Omega(h,t^*) = 0$

SPEC- 1

2.2 Conservation of Mass and Energy

2.2.1 Water Balance, Glacier Water Storage and Glacier Mass Balance

Water Balance, *Glacier Water Storage*, *Glacier Mass Balance*, and *Snowpack, Glacier and Soil Energy* state variables are updated under the principle of conservation of mass and energy. For an HRU *h* equation (1) is re-written and simplified, depending upon the specific case as follows:

CASE 1	$\Omega(h,t^*) = default\ value$
CASE 2	$\Omega(h,t^*) = \Omega(h,t)$
CASE 3	$\Omega(h,t^*) = \Omega(h,t) \cdot \dfrac{A(h,t)}{A(h,t^*)}$
CASE 4	$\Omega(h,t^*) = 0$ $\begin{cases} \Omega_{op}(b,t^*) = \Omega_{op}(b,t) + \Omega(h,t)\dfrac{A(h,t)}{A_{op}(b,t^*)}, & if\ veg\_index(h) = GLACIER \\ \Omega_g(b,t^*) = \Omega_g(b,t) + \Omega(h,t)\dfrac{A(h,t)}{A_g(b,t^*)}, & if\ veg\_index(h) \neq GLACIER \end{cases}$
CASE 5	$\Omega(h,t^*) = 0$

	$$\begin{cases} \Omega_g(b-1,t^*) = \Omega_g(b-1,t) + \Omega(h,t)\dfrac{A(h,t)}{A_g(b-1,t^*)}, & if\ A_g(b-1,t^*) > 0, else \\ \Omega_{op}(b-1,t^*) = \Omega_{op}(b-1,t) + \Omega(h,t)\dfrac{A(h,t)}{A_{op}(b-1,t^*)}, & if\ A_{op}(b-1,t^*) > 0, else \\ \Omega_v(b-1,t^*) = \Omega_v(b-1,t) + \Omega(h,t)\dfrac{A(h,t)}{A_v(b-1,t^*)} & \end{cases}$$

SPEC- 2

It is noted that the current specification for CASE 1 is not very realistic, i.e. although the process of adding a new HRU with state variables defaulting to zero conserves mass it does not necessarily conserve energy; nevertheless, it greatly simplifies the process of state updating. Some order-of-operations dependency has been unavoidable in the current specification. For example, it is conceivable that a new GLACIER HRU could be operated on twice during state updating: once under CASE 1 (initialization when *h* indexes a new GLACIER in band *b*) and again under CASE 4 (e.g. *h* indexes an OPEN HRU in band *b* that disappears as a result of the appearing GLACIER HRU). In this situation CASE 1 must occur before CASE 4, otherwise updates to the GLACIER state under CASE 4 would be overwritten by CASE 1 initialization (hence, the pseudo-code is written to ensure that CASE 1 updating occurs before all other updating).

2.2.2 SNOW_DENSITY

SNOW_DENSITY, which is a function of *SNOW_SWQ* and *SNOW_DEPTH*, is updated using

ALL CASES	$SNOW\_DENSITY(h,t^*) = [SNOW\_SWQ(h,t^*) \cdot 1000]/SNOW\_DEPTH(h,t^*)$	SPEC- 3

where *SNOW_SWQ* and *SNOW_DEPTH* are updated according to SPEC- 2.

2.2.3 GLACIER_WATER_STORAGE

The specification for *GLACIER_WATER_STORAGE*, as it only applies to GLACEIR HRUs, differs slightly from SPEC- 2 (i.e. see CASE 4) as follows

CASE 1	$\Omega(h,t^*) = default\ values$
CASE 2	$\Omega(h,t^*) = \Omega(h,t)$
CASE 3	$\Omega(h,t^*) = \Omega(h,t) \cdot \dfrac{A(h,t)}{A(h,t^*)}$
CASE 4	$\Omega(h,t^*) = 0$

	$\begin{cases} \Omega_{op}(b,t^*) = \Omega_{op}(b,t) + \Omega(h,t)\dfrac{A(h,t)}{A_{op}(b,t^*)}, & if\ veg\_index(h) = GLACIER \\ \Omega_g(b,t^*) = \Omega_g(b,t), & if\ veg\_index(h) \neq GLACIER \end{cases}$
CASE 5	$\Omega(h,t^*) = 0$ $\begin{cases} \Omega_g(b-1,t^*) = \Omega_g(b-1,t) + \Omega(h,t)\dfrac{A(h,t)}{A_g(b-1,t^*)}, & if\ A_g(b-1,t^*) > 0, else \\ \Omega_{op}(b-1,t^*) = \Omega_{op}(b-1,t) + \Omega(h,t)\dfrac{A(h,t)}{A_{op}(b-1,t^*)}, & if\ A_{op}(b-1,t^*) > 0, else \\ \Omega_v(b-1,t^*) = \Omega_v(b-1,t) + \Omega(h,t)\dfrac{A(h,t)}{A_v(b-1,t^*)} \end{cases}$

SPEC- 4

2.2.4 GLACIER_CUM_MASS_BALANCE

The state variable *GLACIER_CUM_MASS_BALANCE*, which applies only to GLACIER HRUs, does not need to be conserved (in a mass sense) and has the following unique specification

CASE 1	$\Omega(h,t^*) = default\ values$
CASE 2	$\Omega(h,t^*) = \Omega(h,t)$
CASE 3	$\Omega(h,t^*) = \Omega(h,t)$
CASE 4	$\Omega(h,t^*) = 0$
CASE 5	$\Omega(h,t^*) = 0$

SPEC- 5

2.2.5 Snow Pack, Glacier and Soil Energy

For state variables grouped under the *Snow Pack, Glacier and Soil Energy* category, variable updating is applied under the principle of conservation of mass and energy. However, the variables *COLD_CONTENT*, *ENERGY_T, ENERGY_TFOLIAGE* and *GLAC_SURF_TEMP* are treated differently, as described in the following paragraphs. Given the updated *SNOW_SURF_TEMP* and *SNOW_SURF_SWQ* for HRU *h*, *COLD_CONTENT* is updated as

ALL CASES	$COLD\_CONTENT(h,t^*) = SNOW\_SURF\_TEMP(h,t^*) \cdot SNOW\_SURF\_SWQ(h,t^*) \cdot CH\_ICE$ where *CH_ICE* is the volumetric heat capacity of ice[b] and *SNOW_SURF_SWQ*(*h*,*t**) = min[*MAX_SURFACE_SWE*, *SNOW_SWQ*(*h*,*t**)]

SPEC- 6

[b] This value for *CH_ICE* is set in the header file vicNl_def.h; currently set to 2100E+03.

	where *MAX_SURFACE_SWE* is the maximum snow water equivalent of the surface layer[c].

For the state variables *GLAC_SURF_TEMP, ENERGY_T* and *ENERGY_TFOLIAGE*, we don't strictly adhere to the conservation of energy principle and simply maintain constant values between state *t* and *t** (continuity principle, see Section 2.4).

2.3 Weighted Assignment

2.3.1 Snow Surface Properties

For variables in the *Snow Surface Properties* category, state variables are updated for CASES 4 and CASE 5 using an area weighting of values at state *t*, given by

CASE 1	$\Omega(h,t^*) = default\ values$
CASE 2	$\Omega(h,t^*) = \Omega(h,t)$
CASE 3	$\Omega(h,t^*) = \Omega(h,t)$
CASE 4	$\Omega(h,t^*) = 0$ $\begin{cases} \Omega_{op}(b,t^*) = \dfrac{\Omega_{op}(b,t)\cdot A_{op}(b,t)\cdot f_{op}(b,t,\mathrm{I}_{SWQ}) + \Omega(h,t)\cdot A(h,t)\cdot f(h,t,\mathrm{I}_{SWQ})}{A_{op}(b,t)\cdot f_{op}(b,t,\mathrm{I}_{SWQ}) + A(h,t)\cdot f(h,t,\mathrm{I}_{SWQ})}, & if\ veg\_index(h) = GLACIER \\ \Omega_{g}(b,t^*) = \dfrac{\Omega_{g}(b,t)\cdot A_{g}(b,t)\cdot f_{g}(b,t,\mathrm{I}_{SWQ}) + \Omega(h,t)\cdot A(h,t)\cdot f(h,t,\mathrm{I}_{SWQ})}{A_{g}(b,t)\cdot f_{g}(b,t,\mathrm{I}_{SWQ}) + A(h,t)\cdot f(h,t,\mathrm{I}_{SWQ})}, & if\ veg\_index(h) \neq GLACIER \end{cases}$
CASE 5	$\Omega(h,t^*) = 0$ $\begin{cases} \Omega_{g}(b-1,t^*) = \dfrac{\Omega_{g}(b-1,t)\cdot A_{g}(b-1,t)\cdot f_{g}(b-1,t,\mathrm{I}_{SWQ}) + \Omega(h,t)\cdot A(h,t)\cdot f(h,t,\mathrm{I}_{SWQ})}{A_{g}(b-1,t)\cdot f_{g}(b-1,t,\mathrm{I}_{SWQ}) + A(h,t)\cdot f(h,t,\mathrm{I}_{SWQ})}, & if\ A_g(b-1,t^*) > 0 \\ \Omega_{op}(b-1,t^*) = \dfrac{\Omega_{op}(b-1,t)\cdot A_{op}(b,t)\cdot f_{op}(b-1,t,\mathrm{I}_{SWQ}) + \Omega(h,t)\cdot A(h,t)\cdot f(h,t,\mathrm{I}_{SWQ})}{A_{op}(b-1,t)\cdot f_{op}(b-1,t,\mathrm{I}_{SWQ}) + A(h,t)\cdot f(h,t,\mathrm{I}_{SWQ})}, & if\ A_{op}(b-1,t^*) > 0 \\ \Omega_{v}(b-1,t^*) = \dfrac{\Omega_{v}(b-1,t)\cdot A_{v}(b-1,t)\cdot f_{v}(b-1,t,\mathrm{I}_{SWQ}) + \Omega(h,t)\cdot A(h,t)\cdot f(h,t,\mathrm{I}_{SWQ})}{A_{v}(b-1,t)\cdot f_{v}(b-1,t,\mathrm{I}_{SWQ}) + A(h,t)\cdot f(h,t,\mathrm{I}_{SWQ})} & \end{cases}$
where $f(h,t,I_{SWQ}) = \mathrm{I}[SNOW\_SWQ(h,t)]$ and $\mathrm{I}(\cdot)$ is the indicator function, such that $\mathrm{I}(X) := \begin{cases} 1 & if\ X > 0 \\ 0 & if\ X \le 0 \end{cases},$ and for *SNOW_MELTING*(h,·) (which must be converted from character to integer)	

[c] This value is set in the header file snow.h; currently set at 0.125 m

$$\Omega(h,\cdot) = \begin{cases} 1 & if\ SNOW\_MELTING(h,\cdot) = "TRUE" \\ 0 & if\ SNOW\_MELTING(h,\cdot) = "FALSE" \end{cases}$$

SPEC- 7

For the state variables *SNOW_LAST_SNOW* and *SNOW_MELTING* (which are integer), SPEC- 7 is further modified as

$$z(t^*) = ceil[\Omega(h,t^*)]$$

and

$$SNOW\_LAST\_SNOW(h,t^*) = z(h,t^*)$$

$$SNOW\_MELTING(h,t^*) = \begin{cases} "TRUE" & if\ z(h,t^*) = 1 \\ "FALSE" & if\ z(h,t^*) = 0 \end{cases}$$

SPEC- 8

2.4 Continuity

2.4.1 Program Terms, Miscellaneous and Deferred Variables

For HRU variables in the *Program Terms* category (and certain variables from other categories), *Miscellaneous* variables (Table 2) and *Deferred* variables (Table 3), state variables are typically updated under the continuity principle. Unless the HRU h is new, values remain constant between state t and t^*. For example, for generic state variable Ω

CASE 1	$\Omega(h,t^*) = default\ values$
CASE 2	$\Omega(h,t^*) = \Omega(h,t)$
CASE 3	$\Omega(h,t^*) = \Omega(h,t)$
CASE 4	$\Omega(h,t^*) = 0$
CASE 5	$\Omega(h,t^*) = 0$

SPEC- 9

2.5 Sanity Check

When water storage state variables are transferred between HRUs (i.e. CASE 4 and CASE 5), we may end up with several non-physically plausible situations. Hence, checks and adjustments need to occur once state updating is completed.

For GLACIER HRUs, perform the following checks and adjustments:

If *SNOW_CANOPY*(*h*,*t**) > 0 then *SNOW_SWQ*(*h*,*t**) += *SNOW_CANOPY*(*h*,*t**) and *SNOW_CANOPY*(*h*,*t**) = 0

SPEC- 10

For OPEN HRUs, perform the following checks and adjustments:

If *SNOW_CANOPY*(*h*,*t**) > 0 then *SNOW_SWQ*(*h*,*t**) += *SNOW_CANOPY*(*h*,*t**) and *SNOW_CANOPY*(*h*,*t**) = 0
If *GLAC_WATER_STORAGE*(*h*,*t**) > 0 then *LAYER_MOIST*[Nlayers-1] += *GLAC_WATER_STORAGE*(*h*,*t**) and *GLAC_WATER_STORAGE*(*h*,*t**) = 0

SPEC- 10

For VEGETAED HRUs, perform the following checks and adjustments:

If *GLAC_WATER_STORAGE*(*h*,*t**) > 0 then *LAYER_MOIST*[Nlayers-1] += *GLAC_WATER_STORAGE*(*h*,*t**) and *GLAC_WATER_STORAGE*(*h*,*t**) = 0

SPEC- 10

3 HRU Specification Summary

The following tables summarize the applicable update specification by state variable. *Mandatory* state variables are summarized in Table 4 and *Miscellaneous* and *Deferred* state variables are summarized in Table 5.

Table 4. Mandatory state variable specification summary

State Variable	Specifications
HRUCELL METADATA	
lat	SPEC- 1
lon	SPEC- 1
GLAC_MASS_BALANCE_INFO	SPEC- 1
GRID_CELL	SPEC- 1
NUM_BANDS	SPEC- 1
NUM_GLAC_MASS_BALANCE_INFO_ TERMS	SPEC- 1
SOIL_DZ_NODE [Nnodes]:	----
SOIL_DZ_NODE [0]	SPEC- 1
SOIL_DZ_NODE [1]	SPEC- 1
⋮	⋮
SOIL_DZ_NODE [Nnodes-1]	SPEC- 1
SOIL_ZSUM_NODE [Nnodes]:	----
SOIL_ZSUM_NODE [0]	SPEC- 1
SOIL_ZSUM_NODE [1]	SPEC- 1
⋮	⋮
SOIL_ZSUM_NODE [Nnodes-1]	SPEC- 1
VEG_TYPE_NUM	SPEC- 1
HRU METADATA	
HRU_BAND_INDEX	SPEC- 1
HRU_VEG_INDEX	SPEC- 1
HRU State Variables	
LAYER_ICE_CONTENT [Nlayers]:	----
LAYER_ICE_CONTENT [0]	SPEC- 2
LAYER_ICE_CONTENT [1]	SPEC- 2
⋮	⋮
LAYER_ICE_CONTENT [Nlayers-1]	SPEC- 2
LAYER_MOIST [Nlayers]	----
LAYER_MOIST [0]	SPEC- 2
LAYER_MOIST [1]	SPEC- 2

State Variable	Specifications
⋮	⋮
LAYER_MOIST [Nlayers-1]	SPEC- 2 & SPEC- 10
HRU_VEG_VAR_WDEW [dist]:	----
HRU_VEG_VAR_WDEW [0]	SPEC- 2
HRU_VEG_VAR_WDEW [1]	SPEC- 2
SNOW_CANOPY	SPEC- 2 & SPEC- 10
SNOW_DEPTH	SPEC- 2
SNOW_DENSITY	SPEC- 3
SNOW_PACK_WATER	SPEC- 2
SNOW_SURF_WATER	SPEC- 2
SNOW_SWQ	SPEC- 2 & SPEC- 10
GLAC_WATER_STORAGE	SPEC- 4 & SPEC- 10
GLAC_CUM_MASS_BALANCE	SPEC- 5
ENERGY_T [Nnodes]:	----
ENERGY_T [0]	SPEC- 9
ENERGY_T [1]	SPEC- 9
⋮	⋮
ENERGY_T [Nnodes-1]	SPEC- 9
ENERGY_TFOLIAGE	SPEC- 9
GLAC_SURF_TEMP	SPEC- 9
SNOW_COLD_CONTENT	SPEC- 6
SNOW_PACK_TEMP	SPEC- 2
SNOW_SURF_TEMP	SPEC- 2
SNOW_ALBEDO	SPEC- 7
SNOW_LAST_SNOW	SPEC- 8
SNOW_MELTING	SPEC- 8
ENERGY_TCANOPY_FBCOUNT	SPEC- 9
ENERGY_T_FBCOUNT [Nnodes]:	----
ENERGY_T_FBCOUNT [0]	SPEC- 9
ENERGY_T_FBCOUNT [1]	SPEC- 9
⋮	⋮
ENERGY_T_FBCOUNT [Nnodes-1]	SPEC- 9
ENERGY_TFOLIAGE_FBCOUNT	SPEC- 9
ENERGY_TSURF_FBCOUNT	SPEC- 9
GLAC_SURF_TEMP_FBCOUNT	SPEC- 9
SNOW_SURF_TEMP_FBCOUNT	SPEC- 9

Table 5. Miscellaneous and deferred state variable specification summary

State Variable	Specifications
Miscellaneous State Variables	
ALL (see Table 2)	SPEC- 9
Deferred State Variables	
ALL (see Table 3)	SPEC- 9