Features of 2.5D Network-on-Chip Router

ΤΗΜΜΥ ΑΠΘ | ΤΟΜΕΑΣ ΗΛΕΚΤΡΟΝΙΚΗΣ & ΥΠΟΛΟΓΙΣΤΩΝ

Χαραλάμπους Νικόλαος | Επιβλέποντες: κ. Παυλίδης Βασίλειος, κ. Τσεκούρας Αριστοτέλης

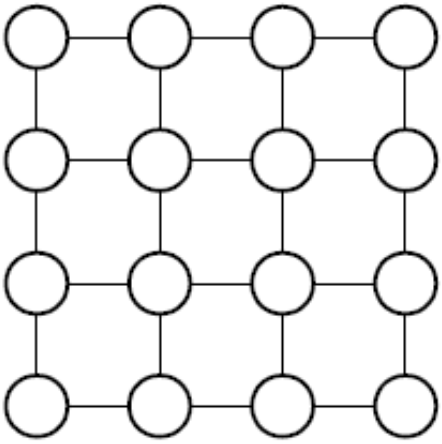
Εισαγωγή

Στο παρόν έγγραφο αναφέρονται επιγραμματικά **οι επιλογές σχεδίασης του 2.5D NoC Router** στα πλαίσια υλοποίησης της διπλωματικής μου εργασίας κατά το ακαδημαϊκό έτος 2025-2026 στον τομέα Ηλεκτρονικής & Υπολογιστών του Τμήματος Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών του ΑΠΘ. Σκοπός της παρόντος εγγράφου είναι **η συγκέντρωση και παρουσίαση των σχεδιαστικών επιλογών** που θα εφαρμοστούν στην υλοποίηση του κυκλώματος σε κώδικα RTL: SystemVerilog. Αξίζει να επισημανθεί πως **είναι πιθανή η διαρκής αλλαγή κι ανανέωση των επιλογών** αυτών στη διάρκεια του ακαδημαϊκού έτους, λόγω των μεταβολών που μπορεί να προκύψουν στο εγχείρημα της υλοποίησης. Αυτές οι αλλαγές, ωστόσο, θα αντανακλώνται στο παρόν έγγραφο μετά από κάθε τυχούσα ανανέωση.

1. Τοπολογία

Η τοπολογία αποτελεί **το πιο θεμελιώδες χαρακτηριστικό** ενός δικτύου διασυνδέσεων και σημαντικό παράγοντα για την απόδοση, την ταχύτητα και την συνολική δομή του δικτύου. Κάποιοι παράγοντες που λαμβάνονται υπόψη, κατά την επιλογή μιας τοπολογίας, είναι η ***διάμετρος*** (μέγιστη απόσταση ανάμεσα σε 2 κόμβους), ο ***βαθμός διασύνδεσης*** κάθε κόμβου, η ***ποικιλία των μονοπατιών (path diversity)***, το ***μέσο μήκος ενός μονοπατιού*** (εξαρτάται από τη διάμετρο), η ***πολυπλοκότητα των διασυνδέσεων*** και η ***προσαρμογή σε διάφορες κλίμακες μεγέθους***. Λαμβάνοντας υπόψη τους παραπάνω παράγοντες, επιλέγουμε για την υλοποίηση του δικτύου την ***τοπολογία πλέγματος (mesh topology)***. Πιο συγκεκριμένα, επιλέγουμε **ένα 2D (δισδιάστατο) 4x4 πλέγμα (mesh) με 16 κόμβους**, το οποίο θα αποτελεί την τοπολογία του δικτύου διασύνδεσης μας. Έχοντας αυτό ως δεδομένο, ένας ***δρομολογητής-router*** αντιστοιχίζεται σε κάθε κόμβο του πλέγματος και πρέπει να σχεδιαστεί με αυτή την προοπτική. Για αυτόν το λόγο, κάθε κόμβος θα έχει **5 θύρες εισόδου-εξόδου** (**4 προς κάθε κατεύθυνση** – Βόρεια, Νότια,

Ανατολικά, Δυτικά **και 1 τοπική προς τον πυρήνα** του στοιχείου με το οποίο συνδέεται ο router).

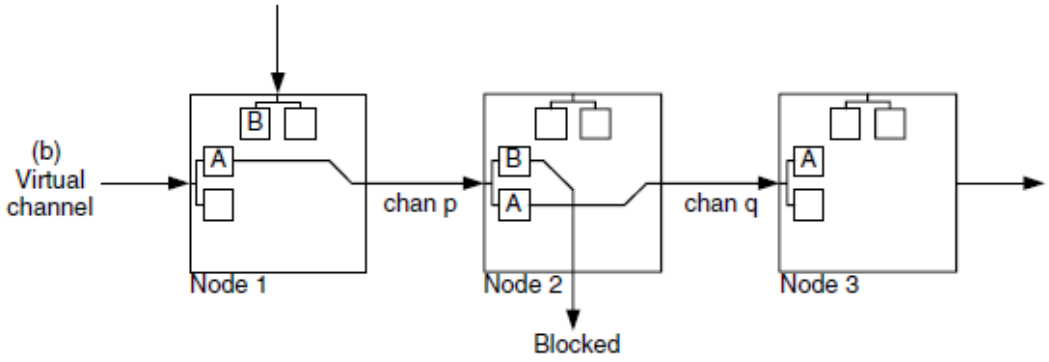


(b) A 4-ary 2-mesh

Εικόνα 1 - 2D 4x4 Τοπολογία Πλέγματος

2. Flow Control (Έλεγχος Ροής)

Το επόμενο βήμα στο πλαίσιο των σχεδιαστικών επιλογών, είναι να αποφασίσουμε με ποιον τρόπο θα πραγματοποιείται η ροή (αποστολή και λήψη) δεδομένων. Υπάρχουν αρκετές πολιτικές ροής ανάλογα με τη χρήση ή μη χρήση τοπικών ***buffers*** (σε κάθε κόμβο) για τη συγκράτηση δεδομένων και αρκετές βελτιωτικές πολιτικές προκειμένου αποφεύγεται το ***blocking*** ανάμεσα στα πακέτα. Αποφασίζοντας να προχωρήσουμε με μια *buffering* πολιτική και λαμβάνοντας υπόψη την δημιουργία *blocking* ανάμεσα στα πακέτα, επιλέγουμε την υλοποίηση ελέγχου ροής με τη χρήση ***εικονικών καναλιών (Virtual***



Εικόνα 2 - Παράδειγμα χρήσης VCs

Channels ~ VCs), τα οποία αποτελούν οντότητες με ξεχωριστούς *buffers* που έχουν τη δυνατότητα να αξιοποιούν από κοινού μια ***φυσική ζεύξη (φυσικό κανάλι)*** έτσι ώστε, σε περίπτωση παύσης αποστολής δεδομένων τα οποία αφορούν ένα συγκεκριμένο ***πακέτο***, να μπορεί να εκμεταλλευτεί τη *φυσική ζεύξη* ένα *εικονικό κανάλι* του οποίου τα δεδομένα μπορούν να προχωρήσουν στο δίκτυο. Πρακτικά, η πολιτική *ροής ελέγχου* με τη χρήση *εικονικών καναλιών*, αποτελεί μια πολιτική σε επίπεδο ***flit*** (καθώς οι *buffers* μπορεί να έχουν μέγεθος μικρότερο από ένα *πακέτο*, δηλαδή μερικών μόνο *flit*) και δημιουργήθηκε ακριβώς για να αντιμετωπιστεί **η αεργία** των *φυσικών ζεύξεων* που οφείλεται σε *blocking* ενός πακέτου από ένα άλλο που χρησιμοποιεί την ίδια ζεύξη αλλά δεν σημειώνει προσωρινά πρόοδο. Ωστόσο έως τώρα δεν διευκρινίστηκε με ποιον τρόπο θα είναι σε θέση να γνωρίζει ένας ***κόμβος αποστολής*** ότι ένας ***κόμβος λήψης*** θα διαθέτει ελεύθερο χώρο στους τοπικούς του *buffer* προκειμένου να υποδεχθεί τα απεσταλμένα *flits*. Για αυτόν τον λόγο χρειαζόμαστε έναν ***μηχανισμό*** ή μια πολιτική ***προς-τα-πίσω πίεσης (backpressure mechanism)***. Ανάμεσα σε πολλούς που χρησιμοποιούνται σε ***buffered flow control***, επιλέγουμε τον ***credit-based backpressure mechanism***. Σύμφωνα με αυτή την πολιτική, κάθε *κόμβος αποστολής* έχει καταχωρημένο σε *registers* τον αριθμό των *ελεύθερων flits* στους *buffer* υποδοχής των *εικονικών καναλιών* στον *κόμβο λήψης*. Αν αυτός ο αριθμός είναι μεγαλύτερος του μηδενός, τότε ο *κόμβος αποστολής* δύναται να αποστέλλει *flits* μέχρι ο *μετρητής* αυτός να μηδενιστεί. Ωστόσο, πρέπει με κάποιον τρόπο οι δύο κόμβοι να επικοινωνούν προκειμένου όταν ελευθερώνεται μια θέση *flit* στον *buffer* λήψης, να ανανεώνεται ο μετρητής του προηγούμενου κόμβου. Για αυτό τον λόγο, ο *κόμβος λήψης*, οποτεδήποτε συμβαίνει αυτό, αποστέλλει ένα ***credit*** (ειδικό *flit ελέγχου* για αυτόν το σκοπό) προς τα πίσω στον *κόμβο αποστολής*, προκειμένου να αυξηθεί κατά 1 η τιμή του μετρητή και να είναι σε θέση να αποστείλει ξανά κάποιο *flit*. Αυτές οι 2 πολιτικές δίνουν ευελιξία στο σχεδιασμό μας, αν και δημιουργούν εν γένει μεγάλη *προς-τα-πίσω σηματοδότηση*, αλλά και θα χρησιμοποιηθούν όπως θα δούμε παρακάτω για την αναγνώριση της ***συμφόρησης*** του δικτύου με τη χρήση της *προς-τα-πίσω πίεσης*. Τέλος, το μέγεθος των *flits*, των *πακέτων* αλλά και ο αριθμός και το βάθος των *buffers*, όπως και ο αριθμός των ***VCs*** ανά ζεύξη, μπορούν να **παραμετροποιηθούν** για να έχουμε μια γενική εποπτεία του κυκλώματος.

Συμπερασματικά, επιλέγουμε :

- **Virtual Channel Flow Control**
- **Credit-Based Backpressure Mechanism**

3. Δρομολόγηση (Routing)

Ο τρόπος με τον οποίο θα αποφασίζεται η διαδρομή των flits και άρα των δεδομένων ενός πακέτου μέσω του δικτύου αφορά το κομμάτι της ***δρομολόγησης***. Υπάρχει μια πληθώρα επιλογών σχετικά με τις πολιτικές δρομολόγησης, οι οποίες επηρεάζουν την ***καθυστέρηση (latency)*** του δικτύου, αλλά και την ***διεκπεραιωτικότητά*** του (***throughput*** ~ βαθμός χρήσης των ζεύξεων) όπως και μια σειρά από άλλους παράγοντες που με τη σειρά τους επηρεάζουν τη συνολική απόδοση του δικτύου. Λαμβάνοντας υπόψη τα παραπάνω και την προσπάθεια αποφυγής ***deadlock*** στο δίκτυο, στην υλοποίηση μας, θα ακολουθήσουμε το θεωρητικό πλαίσιο του ***Duato***. Πιο συγκεκριμένα, θα εισάγουμε την έννοια των ***εικονικών καναλιών διαφυγής (escape VCs)***, και κατόπιν θα χρησιμοποιούμε τα συγκεκριμένα εικονικά κανάλια ως ***υποδίκτυο*** προς αποφυγή ενός πιθανού *deadlock*. Με λίγα λόγια, μπορούμε να έχουμε, π.χ. δύο κλάσεις καναλιών (1 και 2) με την 1 να περιέχει τα εικονικά κανάλια διαφυγής και την 2 να περιέχει τα κανάλια στα οποία γίνεται μια πιο «ελεύθερη» δρομολόγηση. Δηλαδή, επιλέγουμε μέσω των καναλιών της κλάσης 2 να γίνεται μια ***προσαρμοστική ελάχιστη δρομολόγηση (minimal adaptive routing)***, επιλογή σχεδίασης που ευθυγραμμίζεται με τις συνθήκες του *Duato* αλλά δεν αποτελεί απαίτησή του. Η δρομολόγηση αυτή αφενός δεν θα επιτρέπει το ***misrouting*** (αλλαγή πορείας και προσαύξηση απόστασης από τον προορισμό) κι αφετέρου θα είναι προσαρμοστική λαμβάνοντας υπόψη την διαθεσιμότητα των ***downstream VC buffers*** στους γειτονικούς κόμβους (μέσω του μηχανισμού των *credits*), άρα και τη συμφόρηση του δικτύου. Αυτού του είδους η δρομολόγηση θα λαμβάνει αποκλειστικά μέρος στα κανάλια κλάσης 2, ενώ οποτεδήποτε η διαθεσιμότητα είναι περιορισμένη, τα flits μπορούν να μεταβαίνουν από την κλάση 2 στην κλάση 1, αλλά όχι αντίστροφα. Από εκείνη τη στιγμή κι έπειτα θα δρομολογούνται αποκλειστικά μέσω της τελευταίας κλάσης μόνο με ***XY Routing***, όπου έχει αποδειχτεί πως αποτελεί μια ***deadlock-free*** δρομολόγηση σε τοπολογία πλέγματος. Επομένως η κλάση 1 θα επιτελεί την απορρόφηση του δικτύου σε μια *deadlock-free* εγγυημένη δρομολόγηση, οποτεδήποτε η συμφόρηση δεν επιτρέπει την ελάχιστη προσαρμοστική δρομολόγηση στην κλάση 2. Όλα αυτά για την καταπολέμηση του *deadlock*, βασιζόμενοι στην γενικότερη θεωρητική θεμελίωση του *πρωτοκόλλου του Duato*. Επίσης, απαιτώντας η δρομολόγησή μας να είναι ελάχιστη, δημιουργούμε μια ***livelock-free*** δρομολόγηση. Στη θεωρία του *Duato* δεν απαιτείται η μόνιμη παραμονή ενός πακέτου στην *κλάση διαφυγής*. Ωστόσο, στην παρούσα υλοποίηση επιλέγεται η παραμονή στα *escape VCs* μέχρι την παράδοση του πακέτου. Έτσι απλοποιείται ο σχεδιασμός και διατηρείται χωρίς αμφιβολίες το *deadlock-free* χαρακτηριστικό του ***υποδικτύου διαφυγής***. Τέλος, η δρομολόγηση ενός πακέτου δεν βασίζεται σε στατικούς πίνακες διαδρομών, αλλά υπολογίζεται ***δυναμικά*** και ***αλγοριθμικά*** μέσω κάποιου αλγόριθμου

δρομολόγησης, ο οποίος αποφασίζει την επόμενη μετάβαση σε κάποιον γειτονικό κόμβο (***hop***).

Συμπερασματικά, επιλέγουμε :

- **Minimal Adaptive Routing (κλάση 2)**
- **XY Routing μέσω escape VCs (κλάση 1)**

4. Πολιτικές Δέσμευσης εικονικών καναλιών

Διακρίνουμε τις πολιτικές ***(επανα)δέσμευσης*** ενός εικονικού καναλιού σε 2 βασικές κατηγορίες : ***άμεση (aggressive)*** και ***συντηρητική (conservative)***. Ας υποθέσουμε ότι ένας buffer εικονικού καναλιού μπορεί να περιέχει flits από δύο διαφορετικά πακέτα, π.χ. τα A και B. Επίσης, ας υποθέσουμε πως το πακέτο A έχει flits στον *upstream input VC buffer* του τρέχοντος κόμβου, όπως και το πακέτο B άλλωστε, αλλά έχει επιτυχώς δεσμεύσει το *downstream input VC* ενός γειτονικού κόμβου κι επομένως ένα μέρος των flits του πακέτου A βρίσκεται σε αυτόν τον buffer. Η *άμεση επαναδέσμευση* επιτρέπει τη δέσμευση του *downstream input VC* για λογαριασμό του πακέτου B, αμέσως μόλις η ουρά του πακέτου A αφήσει τον *upstream input VC buffer* και μεταβεί στο στάδιο της μεταγωγής. Τότε, αμέσως το πακέτο B μπορεί να δεσμεύσει το εικονικό κανάλι του γειτονικού κόμβου για λογαριασμό του. Βέβαια αυτό μπορεί να προκαλέσει περεταίρω προβλήματα αν το πακέτο A στον επόμενο κόμβο αντιμετωπίσει *blocking*, με συνέπεια να «σταματήσει» η αποστολή και του πακέτου B. Επιπρόσθετα, επειδή το εικονικό κανάλι θα περιέχει και τα 2 πακέτα, αλλά πρακτικά η κατάστασή του θα είναι δεσμευμένη από ένα μόνο πακέτο, θα πρέπει να έχει πραγματοποιήσει σύζευξη με το αντίστοιχο ***εικονικό κανάλι εξόδου (output VC)*** , στο οποίο θέλει να μεταβεί το κάθε πακέτο. Επομένως, αυτό απαιτεί διπλασιασμό των διανυσμάτων κατάστασης του κάθε τρέχοντος ***input VC***, γεγονός που απαιτεί μεγαλύτερη πολυπλοκότητα στον χειρισμό. Ωστόσο αν επιτρέψουμε τη χρήση ενός *input VC buffer* από ένα μοναδικό πακέτο, τότε η χρονική στιγμή που το επόμενο πακέτο μπορεί να δεσμεύσει ένα *downstream input VC*, είναι μόλις η ουρά π.χ. του πακέτου A αφήσει τον *downstream VC buffer* εξ ολοκλήρου άδειο κι έτοιμο για επαναδέσμευση από το πακέτο B, του οποίου η κεφαλή, εκείνη τη χρονική στιγμή θα βρίσκεται στον *upstream input VC buffer* του τρέχοντος κόμβου. Με αυτόν τον τρόπο, μπορεί να έχουμε μεγαλύτερη καθυστέρηση (latency), ωστόσο επιτυγχάνουμε μια πιο ασφαλή δέσμευση με περιθώρια για ευκολότερο χειρισμό. Επομένως, στη σχεδίασή μας, θα προχωρήσουμε με :

- **Συντηρητική επαναδέσμευση εικονικού καναλιού**

5. Πρόσθετες Επιλογές

- **Χρήση ξεχωριστών φυσικών ζεύξεων ανάμεσα σε flits & credits**
- **Χρήση ξεχωριστών κυκλικών buffer ανά εικονικό κανάλι**
- **Βασική αρχιτεκτονική 5×5 Crossbar με Separable Input-first Switch Allocator (Speedup = 1), με προοπτική επέκτασης**
- **Πιθανή χρήση Speculation (βελτίωση κατά την υλοποίηση)**
- **Round-Robin Arbiters**
- **Κλασσική δομή Router με διοχέτευση σταδίων (RC,VA,SA,ST)**

6. Υλοποίηση Σταδίων

Με βάση τις σχεδιαστικές επιλογές που επιλέξαμε στα κεφάλαια Ελέγχου Ροής, Δρομολόγησης και των πρόσθετων επιλογών, προχωράμε στην ανάλυση των σταδίων λειτουργίας ενός ***Virtual Channel*** με ***Credit Backpressure*** Router. Η δομή του δρομολογητή θα αποτελείται από 4 στάδια διοχέτευσης :

- **RC – Routing Computation (Υπολογισμός Δρομολόγησης)**
- **VA – Virtual Channel Allocation (Δέσμευση Εικονικού Καναλιού)**
- **SA – Switch Allocation (Δέσμευση Μεταγωγέα)**
- **ST – Switch Traversal (Μεταγωγή)**

Τα στάδια *RC,VA* εκτελούνται μόνο για το *Head Flit* ενός πακέτου, ενώ τα *SA,ST* για κάθε flit ξεχωριστά.

RC – ROUTING COMPUTATION

1) **<u>Σκοπός:</u>** Ο καθορισμός του συνόλου των **¨νόμιμων¨ θυρών εξόδου** ενός πακέτου **στο τρέχον *router***.
2) Πραγματοποιείται **μόνο** σε *Head Flits*
3) Λαμβάνει ως **είσοδο** :
   a) Τις **τρέχουσες συντεταγμένες** ενός Router
   b) Τις **συντεταγμένες** του **προορισμού**
   c) Την **κλάση εικονικού καναλιού** στην οποία βρίσκεται το flit (*adaptive ή escape VC*)
4) Κανόνες δρομολόγησης
   a) Αν βρίσκεται σε **adaptive VC**
      i) Βρίσκει **όλες τις θύρες** με κριτήριο την **ελάχιστη** δρομολόγηση
   b) Αν βρίσκεται σε **escape VC**
      i) Επιλέγει την μοναδική **θύρα** εξόδου μέσω Deterministic **XY Routing**
5) Παράγει ως **έξοδο** το σύνολο των **υποψήφιων θυρών εξόδου**.

Δηλαδή, αν το *Head Flit* βρίσκεται σε κάποιο εικονικό κανάλι **διαφυγής** (**κλάση των escape VCs**) παραμένει σε αυτή την κλάση και προωθείται μέσω *XY Routing* **μέχρι να παραδοθεί** στον προορισμό. Με λίγα λόγια, επιτρέπονται οι μεταβάσεις:

✓ **adaptive -> adaptive**
✓ **adaptive -> escape**
✓ **escape -> escape**

Αλλά **απαγορεύεται** η μετάβαση :

× **escape -> adaptive**

Η παραπάνω επιλογή διασφαλίζει την αποφυγή ***Deadlock*** **μέσω δρομολόγησης**, ενώ η επιλογή της ελάχιστης δρομολόγησης διασφαλίζει την αποφυγή ***Livelock***.

VA – VIRTUAL CHANNEL ALLOCATION

1) **Σκοπός:** Η **αποκλειστική ανάθεση** ενός *downstream VC* σε **ένα και μόνο** πακέτο.
2) Πραγματοποιείται **μόνο** σε *Head Flits*
3) Εκτελείται **μετά το στάδιο *RC*** και ύστερα από **το φιλτράρισμα με τα κριτήρια επιλεξιμότητας** στα υποψήφια *downstream VCs*
4) **Λειτουργία :**
   a) **Απαρίθμηση** όλων των *downstream VCs* που **αντιστοιχούν** στο σύνολο **των υποψήφιων θυρών εξόδου**
   b) **Φιλτράρισμα** εικονικών καναλιών μέσω των παρακάτω ***κριτηρίων επιλεξιμότητας***. Για να χαρακτηριστεί ένα VC ως **επιλέξιμο**, πρέπει :
      i) Να είναι **ΑΝΕΝΕΡΓΟ**
      ii) Το **πλήθος των credits** στον **μετρητή credits** που διατηρεί ο upstream κόμβος **να είναι > 0**
      iii) Το εικονικό κανάλι να ανήκει σε **επιτρεπόμενη κλάση** για το συγκεκριμένο πακέτο (**adaptive** ή **escape class**)
   c) Αν τα επιλέξιμα downstream εικονικά κανάλια είναι **περισσότερα από ένα** :
      i) Επιλέγουμε **ένα και μοναδικό** μέσω ***Round-Robin*** διαιτησίας **ανάμεσα στα επιλέξιμα** κανάλια
   d) Αλλιώς αν **δεν υπάρχει κανένα κανάλι** με τα παραπάνω κριτήρια επιλεξιμότητας
      i) Το *head flit* **σταματάει** αναμένοντας και **το στάδιο VA επαναλαμβάνεται** στον επόμενο κύκλο ρολογιού
5) Στην περίπτωση **επιτυχούς δέσμευσης**, το επιλεγμένο κανάλι

a) Μεταβαίνει σε **ΕΝΕΡΓΟ**
b) **Κλειδώνει** και **ανήκει** στο συγκεκριμένο πακέτο μέχρι **την αναχώρηση** της **ουράς του πακέτου** (*Tail Flit*) προς τον *μεταγωγέα* (*switch*)

Χρησιμοποιούμε τα παραπάνω (3) κριτήρια επιλεξιμότητας, καθώς :

- Η **διαθεσιμότητα των *credits*** εγγυάται **ελεύθερο χώρο**, με ασφάλεια, στους *buffers*
- Η **κατάσταση** ενός καναλιού εγγυάται **αποκλειστική δέσμευση** για χρήση από **ένα και μόνο πακέτο**
- Η **καθυστέρηση επιστροφής** των ***credits*** **¨κρύβεται¨** (ή γίνεται ανεκτή), αφού απαιτούμε **μόνο ένα** διαθέσιμο credit κι **όχι όλα**

SA – SWITCH ALLOCATION

1) **<u>Σκοπός:</u>** Το στάδιο ***SA*** επιλέγει, **σε κάθε κύκλο ρολογιού**, ποια flits **θα διασχίσουν** τον *μεταγωγέα* (***crossbar switch***), επιλύοντας **τις αντιθέσεις** μεταξύ των εισόδων που απαιτούν την ίδια θύρα εξόδου.
2) Πραγματοποιείται **σε κάθε κύκλο ρολογιού** και αφορά **όλα τα flits** (Head, Body, Tail)
3) **<u>Όροι :</u>**
  a) Κάθε *input VC* μπορεί να **αιτηθεί το πολύ μια θύρα** εξόδου
  b) Κάθε θύρα εξόδου μπορεί να εξυπηρετήσει **το πολύ ένα flit ανά κύκλο**
4) **<u>Αιτήσεις :</u>** Αν για κάποιο *input VC* ισχύουν **οι παρακάτω συνθήκες**, τότε δημιουργείται **αίτηση** προς τη θύρα εξόδου που **σχετίζεται** με το *downstream VC* με το οποίο υπήρξε **σύζευξη** (δέσμευση) μέσω του **σταδίου *VA*** κι έχει αποθηκευτεί στο **διάνυσμα κατάστασης** του τρέχοντος *VC* :
  a) Αν το *input VC* είναι **ΕΝΕΡΓΟ**
  b) **Υπάρχει** flit στην **κεφαλή (head) του *buffer***
  c) Υπάρχει **διαθέσιμο *credit*** για το **αντίστοιχο *downstream VC***
5) **<u>Διαιτησία εισόδου :</u>**
  a) Για κάθε θύρα **εισόδου** επιλέγεται **το πολύ ένα** VC που μπορεί ναι **αιτηθεί κάποια θύρα εξόδου**
  b) Αν αιτούνται **περισσότερα από ένα** VCs **της ίδιας θύρας εισόδου**, αποφασίζεται **ένα και μοναδικό** μέσω ***Round-Robin arbitration***
  c) Υπάρχει πιθανότητα κάποια **θύρα εισόδου** να **μην έχει αιτήσεις** από ***input VCs*** λόγω **έλλειψης flits** στους buffers ή κάποιου ***credit stall***
6) **<u>Διαιτησία εξόδου :</u>**
  a) **Για κάθε θύρα εξόδου** επιλέγεται **το πολύ ένα** εικονικό κανάλι εισόδου από αυτά που **κέρδισαν τον προηγούμενο γύρο διαιτησίας** και **αιτούνται** τη θύρα αυτή

b) Αν υπάρχουν **πολλαπλές αιτήσεις** από **διαφορετικές θύρες εισόδου** προς την **ίδια θύρα εξόδου**, επιλέγεται **ένα και μοναδικό *input VC*** μέσω ***Round-Robin διαιτησίας***
c) Ο ***arbiter*** κάθε θύρας εξόδου εξετάζει **τις αιτήσεις** που **αφορούν μόνο** τη συγκεκριμένη θύρα
d) Υπάρχει πιθανότητα, μια θύρα εξόδου να μείνει **άεργη** (**μικρότερο *throughput***) εξαιτίας **έλλειψης αιτήσεων** προς αυτήν

7) **Έξοδος :**
a) **Το πολύ ένα** σήμα ***επίτρεψης (grant)*** προς το *VC* που νίκησε ταυτόχρονα το ***input*** και το ***output arbitration***
b) Τα ***σήματα επίτρεψης (grants)*** χρησιμοποιούνται από ***το στάδιο ST*** για τη **διέλευση** των *flits* μέσω του **μεταγωγέα flit** (*crossbar switch*)

Είναι σημαντικό να αναφερθεί πως, επειδή το στάδιο *SA* εκτελείται **για κάθε flit ξεχωριστά**, **κι όχι μόνο στο head flit** ενός πακέτου, αυτό αποτελεί μια **ειδοποιό διαφορά** σε σχέση με τα στάδια *RC* & *VA*.

ST – SWITCH TRAVERSAL

1) **Σκοπός :** Η **φυσική διέλευση των fits** μέσω του **μεταγωγέα**, σύμφωνα με **τα σήματα επίτρεψης** (***grants***) που παρήχθησαν στο ***στάδιο SA***
2) **Λειτουργία :**
a) Εκτελείται σε **κάθε κύκλο** ρολογιού
b) Για κάθε ***grant***, το **αντίστοιχο flit** προωθείται από τη θύρα **εισόδου** στην **προοριζόμενη** θύρα **εξόδου**
c) Ο μεταγωγέας επιτρέπει τη **διέλευση ενός flit ανά κύκλο** σε **κάθε θύρα εξόδου**
3) **Ενημέρωση Κατάστασης :** Κατά την **επιτυχή μεταγωγή** (διέλευση) ενός flit :
a) **Αφαιρείται** το συγκεκριμένο **flit** από τον ***input buffer***
b) **Μειώνεται κατά 1**, στο **διάνυσμα κατάστασης**, ο **μετρητής credits** του ***upstream*** κόμβου για το ***downstream input VC***
c) Αν το flit αυτό, είναι ***Tail flit*** :
i) Το ***upstream*** εικονικό κανάλι **αποδεσμεύεται** και μεταβαίνει σε **ΑΝΕΝΕΡΓΟ**
4) **Μηχανισμός Credits :** Μόλις πραγματοποιηθεί η **μεταγωγή** κάποιου flit, δηλαδή η **αναχώρησή** του από τον *input buffer* :
a) Πραγματοποιείται **επιστροφή ενός credit** προς την **αντίθετη κατεύθυνση**
b) Το ***στάδιο ST* επανεκτελείται** στον **επόμενο κύκλο**, **χωρίς** να περιμένει **να φτάσει το credit** που έστειλε στον **upstream κόμβο**

7. Σχεδιαστικές Παράμετροι

Η υλοποίηση θα πραγματοποιηθεί σε πρώτο στάδιο με συγκεκριμένες τιμές σχεδιαστικών παραμέτρων. Οι τιμές αυτές, επιλέχθηκαν με σκοπό την αναπαράσταση μιας ρεαλιστικής & ακαδημαϊκά συμβατής, βάσης για σχεδίαση router, στοχεύοντας παράλληλα σε υλοποιήσιμο βαθμό της πολυπλοκότητας σχεδίασης και επαλήθευσης.

- #Θύρες = 5 (6?)
- Μέγεθος Flit = 64 bit
- MAX #Flits ανά πακέτο = 4 flits
- #VCs = 2 (escape & adaptive)
- Μέγεθος VC buffer = 4 flits
- #Θύρες Crossbar = 5(6?)

8. Διασαφήνιση Θυρών εισόδου & εξόδου

Τα ***modules*** που κατασκευάζονται διαθέτουν θύρες σημάτων εισόδου κι εξόδου, τα οποία επιτελούν ένα συγκεκριμένο έργο κι επεξεργάζονται ή μεταβιβάζονται προς/από το υπάρχον module. Σε αυτό το κεφάλαιο θα εξετάσουμε ξεχωριστά τα σήματα των θυρών εισόδου κι εξόδου του κάθε module, καθώς και θα παραθέσουμε μια συνοπτική επισκόπηση του έργου και το σκοπού που επιτελεί το κάθε σήμα.

module circular_buffer

Όνομα	Κατεύθυνση	Τύπος	Αριθμός bits	Περιγραφή
data_i	Input	flit_t	64	Εισερχόμενο προς εγγραφή flit
read_i	Input	logic	1	Εντολή για ανάγνωση & εξαγωγή flit
write_i	Input	logic	1	Εντολή για εγγραφή flit
rst	Input	logic	1	Εντολή Reset
clk	Input	logic	1	Ρολόι
data_o	Output	flit_t	64	Το τελευταίο flit το οποίο

				εξήχθη (ανανεώνεται στο posedge)
peek_o	Output	flit_t	64	Flit αποθηκευμένο στην κεφαλή του buffer (έτοιμο για εξαγωγή) (ανανεώνεται συνδυαστικά)
is_full	Output	logic	1	Σήμα πληρότητας buffer
is_empty	Output	logic	1	Σήμα άδειου buffer

module vc_buffer

Όνομα	**Κατεύθυνση**	**Τύπος**	**Αριθμός bits**	**Περιγραφή**
data_i	Input	flit_t	64	Εισερχόμενο προς εγγραφή flit
read_i	Input	logic	1	Εντολή ανάγνωσης & εξαγωγής flit
write_i	Input	logic	1	Εντολή εγγραφής flit
vc_new_i	Input	logic	(1 - default)	ID δεσμευμένου downstream VC (έρχεται από VC allocator)
vc_valid_i	Input	logic	1	Σήμα επιτυχούς δέσμευσης downstream VC (έρχεται από VC allocator)
out_port_i	Input	port_t	3	ID θύρας δρομολόγησης

				εξόδου (έρχεται από RC unit)
rst	Input	logic	1	Εντολή Reset
clk	Input	logic	1	Ρολόι
data_o	Output	flit_t	64	Το τελευταίο flit το οποίο εξήχθη (ανανεώνεται στο posedge)
peek_o	Output	flit_t	64	Flit αποθηκευμένο στην κεφαλή του buffer (έτοιμο για εξαγωγή) (ανανεώνεται συνδυαστικά)
is_full_o	Output	logic	1	Σήμα πληρότητας buffer
is_empty_o	Output	logic	1	Σήμα άδειου buffer
out_port_o	Output	port_t	3	ID δρομολόγησης θύρας εξόδου (υπολογίζεται στο RC unit)
vc_request_o	Output	logic	1	Σήμα αιτήματος δέσμευσης downstream VC (πάει στον VC allocator)
switch_request_o	Output	logic	1	Σήμα αιτήματος δέσμευσης μεταγωγέα (πάει στον switch allocator)
vc_allocatable_o	Output	logic	1	Σήμα ελεύθερου VC για επόμενη δέσμευση

				(πάει στον VC allocator)
downstream_vc_o	Output	logic	(1 – default)	ID δεσμευμένου downstream VC (υπολογίζεται από VC allocator)
error_o	Output	logic	1	Σήμα σφάλματος